

# A Novel Zero-Trust Identity Framework for Agentic AI: Decentralized Authentication and Fine-Grained Access Control

Ken Huang<sup>\*</sup>, Vineeth Sai Narajala<sup>†</sup>, John Yeoh<sup>‡</sup>, Json Ross<sup>§</sup>, Mahesh Lambe<sup>¶</sup>,  
Ramesh Raskar<sup>||</sup>, Youssef Harkati<sup>\*\*</sup>, Jerry Huang<sup>††</sup>, Idan Habler<sup>‡‡</sup>, Chris Hughes<sup>x</sup>

<sup>\*</sup>Fellow and Co-Chair, CSA AI Safety Working Groups

<sup>†</sup>Application Security Engineer, Amazon Web Services

<sup>‡</sup>Chief Scientific Officer, EVP, Cloud Security Alliance

<sup>§</sup>Product Security Principal, Salesforce

<sup>¶</sup>MIT NANDA Coauthor, Stanford GSB alumnus

<sup>||</sup>Professor, MIT; Founder, MIT Project Nanda for Agentic Web

<sup>\*\*</sup>Co-founder and CTO, BrightOnLABS

<sup>††</sup>Researcher, The University of Chicago

<sup>‡‡</sup>Independent Researcher

<sup>x</sup>Founder, Resilient Cyber

**Abstract**—Traditional Identity and Access Management (IAM) systems, primarily designed for human users or static machine identities via protocols such as OAuth, OpenID Connect (OIDC), and SAML, prove fundamentally inadequate for the dynamic, interdependent, and often ephemeral nature of AI agents operating at scale within Multi Agent Systems (MAS) – a computational system composed of multiple interacting intelligent agents that work collectively.

This paper posits the imperative for a novel Agentic AI - IAM framework: We deconstruct the limitations of existing protocols when applied to MAS, illustrating with concrete examples why their coarse-grained controls, single-entity focus, and lack of context-awareness falter. We then propose a comprehensive framework built upon rich, verifiable Agent Identities (IDs), leveraging Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs), that encapsulate an agent’s capabilities, provenance, behavioral scope, and security posture.

Our framework includes an Agent Naming Service (ANS) for secure and capability-aware discovery, dynamic fine-grained access control mechanisms, and critically, a unified global session management and policy enforcement layer for real-time control and consistent revocation across heterogeneous agent communication protocols. We also explore how Zero-Knowledge Proofs (ZKPs) enable privacy-preserving attribute disclosure and verifiable policy compliance.

We outline the architecture, operational lifecycle, innovative contributions, and security considerations of this new IAM paradigm, aiming to establish the foundational trust, accountability, and security necessary for the burgeoning field of agentic AI and the complex ecosystems they will inhabit.

**Index Terms**—Agentic AI, Identity Management, Access Control, Multi-Agent Systems, Decentralized Identifiers, Verifiable Credentials, Zero-Knowledge Proofs, AI Security, Zero Trust, IAM, FGAC.

## I. INTRODUCTION

Do we need a new approach for Agentic AI Identity Management? The failure to address the unique identity

challenges posed by AI agents operating in Multi-Agent Systems (MAS) could lead to catastrophic security breaches, loss of accountability, and erosion of trust in these powerful technologies. For instance, without robust agent-specific IAM, a compromised autonomous agent in a financial system could cascade unauthorized transactions, or a swarm of interacting agents in critical infrastructure could be manipulated with devastating consequences. In this Cloud Security Alliance paper [1], We listed initial reasons and approach. This paper expanded on our previous paper and proposed a more robust approach. (ABAC [2], PBAC [3], JIT [4], [5])

The core problem this current paper addresses is the fundamental mismatch between existing IAM paradigms (e.g., OAuth 2.0, OpenID Connect (OIDC), SAML) and the unique characteristics of AI agents in MAS. These agents exhibit autonomy, ephemerality, dynamically evolving capabilities, complex trust relationships, and operate at an unprecedented scale. Their actions carry direct consequences, demanding robust accountability. Delegated authority can cascade through multiple agents, obscuring responsibility if not managed appropriately. The European Union’s AI Act [6] and similar regulatory initiatives underscore the growing societal demand for transparency, accountability, and human oversight in AI systems, making robust agent IAM an unavoidable prerequisite.

In a Wall Street Journal article, on May 17, 2025, Rosenbush [7] discusses the challenges AI agents encounter in accessing applications, APIs, and websites, emphasizing the need for new authentication methods beyond traditional human-centric options.

Inspired by preliminary discussions on IDs for AI systems [8], this paper examines the limitations of current IAM protocols in MAS settings and illustrate, through concrete examples, how their coarse-grained permissions, single-entity assumptions,

limited inclusion of Non-Human Identities (NHIs) and lack of contextual adaptability fall short and propose a need for a new, holistic Agentic AI IAM framework. We contend that merely adapting existing protocols is insufficient. Instead, a purpose-built approach is required, one that redefines agent identity, incorporates novel cryptographic primitives, and establishes new mechanisms for discovery, layered authentication, access control, and real-time policy enforcement tailored to the agentic paradigm.

This paper makes the following contributions:

- It critically analyzes the inadequacies of traditional IAM protocols (OAuth, OIDC, SAML) in the context of MAS, providing concrete examples of their failure points.
- It defines the essential components of a rich, verifiable, and dynamic AI Agent Identity (ID), leveraging Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs).
- It proposes a layered Agentic AI IAM architectural framework incorporating DIDs, VCs, Zero-Knowledge Proofs (ZKPs), an Agent Naming and Discovery Service (ANS) [9], dynamic access control models, and a novel unified global session management and policy enforcement layer.
- It details how this framework addresses the lifecycle of agent IAM, from identity creation and attestation to runtime authorization, logging, monitoring, and incident response.
- It compares centralized, decentralized, and federated deployment models for this framework, offering guidance on their applicability, and analyzes security considerations using the MAESTRO framework.

The remainder of this paper is structured as follows: Section II elaborates on the imperative for a new agentic IAM paradigm by dissecting the limitations of traditional IAM. Section III defines the multifaceted nature of an AI agent's identity. Section IV presents the proposed Agentic AI IAM framework architecture. Section V discusses the operational use cases of Agent IDs within this framework. Section VI analyzes deployment models and governance. Section VII details security considerations. Section VIII highlights the innovative contributions. Section IX discusses future work, and Section X concludes.

## II. THE IMPERATIVE FOR A NEW AGENTIC IAM PARADIGM

The rise of MAS necessitates a fundamental rethinking of how we manage identity and access. While traditional IAM protocols have served well for human-centric and simpler machine-to-machine interactions via service accounts, their core assumptions and mechanisms break down when faced with the complexities of autonomous, interacting AI agents.

### A. Revisiting Traditional IAM

Protocols like OAuth 2.0 [10], OpenID Connect (OIDC) [11], and SAML [12] are ubiquitous for authentication and authorization. Alongside these, foundational enterprise protocols such as Kerberos [13] for domain authentication and LDAP [14] for directory services, as well as comprehensive

cloud identity solutions like Microsoft Entra ID, form the backbone of current identity management for human users and traditional IT systems. Let us discuss their utility and, more importantly, their profound insufficiencies for MAS.

1) *Lingering Utility for Constrained Scenarios:* In limited contexts, particularly involving single agents or direct human-to-agent platform interactions, these traditional protocols and systems can still play a role, primarily in managing the human interface to agentic systems or bootstrapping initial agent context:

- **Human Authentication to Platforms:** OIDC and SAML for Web/Federated Access: A human user authenticating to an AI agent deployment platform via OIDC or SAML is a standard use case. This is often federated through broader cloud identity solutions like Microsoft Entra ID, which can manage both cloud-native and synchronized enterprise identities. For instance, a developer logging into an AI orchestration platform would use their enterprise OIDC provider. Kerberos for Enterprise Internal Access: Within many corporate networks, Kerberos remains the primary mechanism for authenticating human users to internal services and platforms. A developer or operator might authenticate to their workstation and subsequently to an agent management console using Kerberos.
- **Deriving Initial Agent Context and Attributes:** LDAP as an Attribute Source: Enterprise LDAP directories (such as those underpinning Active Directory, often managed or federated by Microsoft Entra ID in hybrid environments) serve as authoritative sources for user attributes and group memberships. This information can be used by an organization to issue initial Verifiable Credentials (VCs) to an agent, attesting to its ownership, departmental affiliation, or preliminary set of permissions derived from the human deployer's context. The platform may then spawn agents that initially operate under a context derived from this human user's authenticated session. The platform then creates an agent, `mcp-dev-agent`, which might initially inherit some basic permissions tied to the developer's identity (sourced via OIDC, SAML, or Kerberos, with attributes potentially enriched from LDAP) to access specific code repositories and documentation systems. OAuth 2.0 for Simple Delegated Access by a Single Agent: An AI agent acting as a confidential client can use OAuth 2.0 to access a resource server on behalf of a human user who has granted explicit consent. This mirrors traditional third-party application access. If `mcp-dev-agent` needs to retrieve additional project context using Model Context Protocol (MCP) [15]–[17] to better understand the developer's codebase, it would go through a standard OAuth 2.0 flow, obtaining an access token scoped specifically to read project documentation and code structures that the developer has authorized.
- **NHI Tasks and Automations:** NHIs can inherit access permissions from the human who deployed them. These identities, while non-autonomous and task-specific, are typically predictable, constrained, and managed through

traditional IAM protocols. Service Accounts and OAuth 2.0: Traditional NHIs like service accounts often rely on OAuth 2.0 client credentials flows to authenticate to cloud APIs or internal services. These flows are compatible with existing identity governance platforms, though they lack behavioral awareness and session integrity. Secrets and Certificates as Surrogate Authentication: Static secrets and certificates issued through PKI or secret management systems are effective in authentication but lack real-time behavior verification, traceability, and support in dynamic environments. Role-Based Access Tied to Humans: NHIs in many organizations are indirectly managed by assigning them roles or permissions derived from human owners or creators (e.g., LDAP group inheritance or IAM role mapping). This makes sense for simple automation tools but fails in autonomous systems.

However, these scenarios typically involve a single, well-defined agent acting in a relatively static role, often directly tethered to a human user's session or a pre-configured machine identity derived from these traditional IAM systems. The complexities, and the breakdown of these approaches, arise when multiple agents interact autonomously, as detailed next.

2) *Fundamental Insufficiencies for Multi-Agent Systems (MAS)*: The dynamic, decentralized, and deeply interconnected nature of MAS exposes critical flaws in traditional IAM:

- **Coarse-Grained and Static Permissions:** OAuth and SAML primarily rely on pre-defined scopes or roles that are often too broad and static for the fluid operational needs of AI agents. Agents in MAS frequently require granular, task-specific permissions that can change dynamically based on context, mission objectives, or real-time data analysis [18].

*Example:* Consider a disaster response MAS. Agent-Search (locates survivors via `drone_feed_api`) might initially need read-only access to map data (`map.read`) and drone telemetry (`drone.telemetry.read`). Upon finding a survivor, it might need to delegate a task to Agent-MedicalDispatch (coordinates `medical_resources_api`), which then requires access to `medical_assets.request` and `hospital_availability.query`. Agent-Search might then also need to alert Agent-Logistics (manages `supply_chain_api`) about resource needs, requiring `supply.request` permissions. In this example, traditional OAuth scopes (`read_all_data`, `manage_all_resources`) would lead to massive over-privileging, while re-authenticating for every micro-permission change is untenable.

- **Single-Entity Focus vs. Complex Delegations:** These protocols are architected around a single authenticated principal (user or application) [19]. They struggle to model and secure complex delegation chains where an agent might spawn sub-agents, or where an agent acts on behalf of multiple principals simultaneously (e.g., a user and an organization).

*Example:* A user (`userAlice_DID`) delegates a financial planning task to Agent-Planner (`agentPlanner_DID`).

Agent-Planner determines it needs specialized market analysis and spawns Agent-MarketAnalyst (`agentMarketAnalyst_DID`) and tax optimization from Agent-TaxOptimizer (`agentTaxOptimizer_DID`). How is `userAlice_DID`'s authority securely and granularly passed from Agent-Planner to its sub-agents? Does Agent-MarketAnalyst inherit all of Agent-Planner's (and thus `userAlice_DID`'s) permissions, or just the bare minimum for market data access? OAuth's delegation (e.g., token exchange) is typically designed for simpler (often one-hop) scenarios and doesn't provide a clear, auditable chain of fine-grained delegated authority. As a result, using the OAuth model, accountability becomes blurred: if Agent-TaxOptimizer accesses unauthorized client data, is Agent-Planner or `userAlice_DID` responsible?

- **Limited Context Awareness:** Traditional IAM decisions are largely based on static roles or scopes, with minimal understanding of the runtime context, agent intent, or associated risk level [20]. Access is often granted at the beginning of a session and persists, irrespective of evolving circumstances.

*Example:* An inventory management agent (Agent-Inventory) has permissions to update stock levels (`inventory.write`). If it attempts to update stock levels for a product that has been recalled (an environmental condition) or tries to zero out all inventory (anomalous behavior), traditional IAM systems typically lack the contextual awareness to flag this as suspicious or dynamically restrict the permission.

- **Scalability Issues with Token/Session Management:** For organizations deploying hundreds or thousands of (potentially ephemeral) agents, each potentially interacting with numerous services, the volume of authentication events and tokens can overwhelm traditional IAM infrastructure [21]. Managing issuance, validation, and especially revocation of a massive number of short-lived tokens becomes an operational nightmare.

*Example:* An e-commerce platform deploys thousands of personalized shopping assistant agents for users. In this example, each agent might exist for only a few minutes. The overhead of frequent, secure token management with traditional protocols is a significant barrier.

- **Dynamic Trust Models & Inter-Agent Authentication:** Agents in MAS often need to authenticate and authorize each other, potentially across organizational boundaries, without a pre-existing, universal trust fabric. OAuth and SAML assume a hierarchical trust model (user trusts IdP, SP trusts IdP). Peer-to-peer trust establishment between autonomous agents from different trust domains is not natively supported [22].

*Example:* Agent-Alpha from "AlphaCorp" needs to request data processing from Agent-Beta from "BetaInc." How do they mutually authenticate? How does Agent-Beta verify Agent-Alpha's capabilities or authorization to request this specific processing without resorting to cumbersome pre-shared secrets or custom API key mechanisms for every

pair of interacting agents?

- **NHI Proliferation and Management Crisis:** Each autonomous agent may require NHIs for numerous APIs, databases, and services, leading to an exponential growth in secrets that must be securely stored, rotated, and managed [23]. This "secret sprawl" increases the attack surface significantly.

*Example:* A single supply chain optimization agent might need API keys for: a shipping provider, a warehousing system, a customs declaration service, an internal ERP.

- **Global Logout/Revocation Complexity:** If an agent is compromised or its task is complete, ensuring its access rights and sessions are immediately and comprehensively revoked across all systems it interacts with is a major challenge with traditional, often session-based protocols [24]. Fragmented revocation mechanisms can leave lingering access.

*Example:* An agent Agent-DataAggregator has active sessions with three different microservices using OAuth tokens. If the agent is detected as compromised, revoking its token at the authorization server is step one. Ensuring each microservice immediately invalidates its session based on that token, especially if they cache permissions, requires a coordinated effort not always inherent in standard OAuth.

#### *B. Unique Challenges Posed by Agentic AI in MAS Further Exacerbating IAM Deficiencies*

Beyond the protocol mismatches, the very nature of agentic AI introduces further complexities:

- **Autonomy and Potential Unpredictability:** Agents with high degrees of autonomy can make decisions that were not explicitly programmed, potentially leading to unforeseen interactions or resource access attempts that challenge static policy definitions.
- **Ephemerality and Dynamic Lifecycles:** Agents can be created, cloned, and destroyed rapidly based on demand. Managing identities and access for such transient entities with persistent credentials is risky and inefficient. An "ephemeral authentication" approach is needed [25].
- **Evolving Capabilities and Intent:** Agents, particularly those incorporating online learning, can adapt their behavior and even their goals over time. An IAM system must be able to accommodate or constrain such evolution.
- **Need for Verifiable Provenance and Accountability:** Tracing actions back to a specific agent instance, understanding its decision-making process (especially if it involved other agents or tools), and ensuring non-repudiation is crucial for trust and forensics.
- **Preventing Autonomous Privilege Escalation:** A sophisticated agent might probe its environment or interact with management APIs to grant itself higher privileges if not carefully constrained. Additionally, agents may interact with each other in a way that leads to privilege escalation through their combined actions, in a manner similar to collusion among humans.

- **Risks of Over-Scoping Access and Permissions:** Agents will actively explore and utilize every permission available to them. This pervasive behavior demands a shift to tightly scoped, task-specific, and context-based access controls to prevent over-privilege and unintended access to sensitive data and environments.
- **Secure and Efficient Cross-Agent Communication & Collaboration:** As agents increasingly form ad-hoc teams or workflows, the need for secure, low-overhead authentication and authorization between them becomes paramount.
- **Actions Taken May Not Directly Correlate to Human Requests:** As agents are given increasing autonomy and reasoning capabilities, the direct tie between a given human goal and actions taken by any particular agent may no longer exist. For example, a management agent may decide to request a worker agent to use a tool based on its own reasoning, rather than at the specific request of a human. An IAM system must be able to discern between when an action is taken at the direct request of a human, and when it is the result of an agentic decision.

These challenges collectively demonstrate that a reactive, bolt-on approach to agent IAM is insufficient. A proactive, purpose-built architectural framework is imperative to harness the power of MAS securely and responsibly. Traditional IAM systems provide a shaky foundation for the towering edifice of interconnected, autonomous AI agents.

### III. DEFINING THE AGENT IDENTITY (AGENT ID) FOR A NEW ERA

To address the challenges of Agentic AI IAM, we must first redefine what constitutes an "identity" for an AI agent. It transcends a simple API key or a username/password. An Agent ID in a MAS context must be a rich, verifiable, dynamic, and cryptographically secured profile that serves as the foundation for trust, access control, and accountability.

#### *A. What Constitutes an AI Agent's Identity? Beyond Static Identifiers*

An AI agent's identity is not merely a label but a comprehensive digital representation that captures its origin, purpose, capabilities, behavior, relationships, and attestations. Agent IDs represent a subset of NHIs that are autonomous, goal-driven, and context-aware. However, to function effectively, agents also rely on or control other types of NHIs (i.e., API tokens, service accounts, workload identities access external resources, execute API calls, or authenticate to services). Agent IDs must be uniquely distinguishable, even when agents are cloned or operate ephemerally, and it must support verification of claims made by or about the agent. We define an "instance" of an AI agent as a runtime instantiation of an agent's software and model, combined with its unique state, memory, and interaction history at a given point in time. Table I outlines different identity models for agents based on their lifespan, origin, and hierarchical relationships, highlighting how unique identifiers support traceability and attribution.



TABLE I  
AGENT IDENTITY MODELS IN MULTI-AGENT SYSTEMS

Agent Type	Description
Persistent Agents	For long-lived agents, the ID provides a continuous thread of identity across sessions, state changes, and even restarts, as long as core attributes and memory persist.
Ephemeral Agents	Each execution of a short-lived, task-specific agent constitutes a new instance with a unique (potentially derived) ID, ensuring that its actions are distinctly attributable, even if its lifespan is mere seconds.
Agent Copies/Forks	A copied or forked agent becomes a distinct instance with its own unique ID, diverging from its parent over time. The relationship to the parent (provenance) should be part of its identity.
Hierarchical Agents	Sub-agents spawned by a parent agent are separate instances, each with a unique ID, but with a verifiable link (e.g., via a Verifiable Credential) back to the parent, enabling traceable delegation.

### B. Essential Components of an Agent ID

The proposed Agent ID, ideally anchored by a Decentralized Identifier (DID) [26], should encapsulate a wide array of information within its associated DID Document and through Verifiable Credentials (VCs) [27], [28]. These components allow for a holistic representation:

#### (A) Cryptographic Anchor & Verifier:

- **Decentralized Identifier (DID):** The globally unique, persistent, and resolvable root identifier (e.g., did:example:agent123). The DID method dictates how it's registered and resolved.
- **Associated Cryptographic Key Pairs:** Public/private key pairs linked to the DID, specified in the verification-Method section of the DID Document. These are used for signing agent actions, encrypting communications, and authenticating the agent when it presents its DID.
- **DID Document Service Endpoints:** Pointers to services associated with the agent, such as its communication endpoints or a profile service.

#### (B) Core Attributes & Metadata (Often in DID Document or VCs):

- **Creator/Deployer/Owner/Controller:** DIDs or other identifiers of the entities responsible for the agent's creation, operation, and governance.
- **Agent Software Version & Model Information:** Cryptographic hash of the agent's core model parameters and software version. We recommend the use of FIPS-approved SHA-3 family hash functions (SHA3-224, SHA3-256, SHA3-384, and SHA3-512) to ensure strong cryptographic security.
- **Timestamps:** Creation date, last update, expected expiry (for ephemeral IDs).
- **Dependencies(Optional):** A list of critical software components, libraries, or other agent services that this agent relies upon. This is optional metadata and a normative reference to AIBOM is preferred way to

define the dependencies.

- **Training Information (Optional):** Details about the datasets, methods, and environment used to train the agent's underlying model.
- **Lifecycle Status:** Current state (e.g., active, suspended, revoked, archived).

#### (C) Capabilities, Scope, and Behavior (Crucial for Access Control & Trust):

- **Formal Scope of Behavior:** A machine-readable definition of the agent's intended tasks, operational domains, and interaction boundaries.
- **Decision-Making Capabilities:** Details on the agent's model type, primary reasoning methods, and key behavioral parameters.
- **Toolset:** An explicit, verifiable list of the tools, APIs, or other agents it is authorized to use.
- **Expected Outcomes & Limitations:** Definition of intended successful outcomes and known failure modes or limitations.

#### (D) Operational & Security Parameters:

- **Communication Protocols Supported:** Specification of protocols the agent can use.
- **Security Properties Attested:** Claims about security features.
- **Compliance Information:** VCs asserting compliance with relevant regulations.
- **Update Mechanism:** Information on how the agent's software, model, or DID Document can be securely updated.

#### (E) Verifiable Credentials (VCs): The Key to Dynamic Attributes and Trust: VCs are digitally signed attestations about an agent, issued by a trusted entity. Usually, trusted entities are government agencies or big IT companies acting as Certification Authorities. Agents can hold and present these VCs to prove specific attributes or authorizations.

- **Role VCs:** "DisasterResponseCoordinatorRole".
- **Capability VCs:** "CertifiedToUse\_MedicalImagingAI\_v3".
- **Reputation VCs:** "TrustedCollaborator\_Score\_95\_Percentile\_from\_CommunityX".
- **Provenance VCs:** "SpawnedBy\_did:example:parentAgent789\_at\_TimestampZ".

### C. Agent ID Ownership and Control

A cornerstone of this new IAM paradigm is the principle of Self-Sovereign Identity (SSI) applied to agents.

- **Agent (or its designated controller) as Holder:** The agent itself or its designated controller holds the private keys associated with its DID and manages its VCs.
- **Controller:** The entity ultimately responsible for the agent.
- **Decoupling from Issuers and Verifiers:** The agent's identity is not solely dependent on a single centralized identity provider.

This model moves away from centrally managed identities, empowering the agent/controller with greater control and portability.

#### *D. ID Generation, Assignment, and Lifecycle Management: From Birth to Revocation*

Managing the lifecycle of these rich Agent IDs is crucial.

- **Initial ID Generation and Assignment:**

- Centralized Platform Issuance: In enterprise settings, a platform might generate a DID for an agent upon deployment.
- Decentralized/Self-Issuance: An agent or its controller can generate its own DID using a suitable DID method.
- Initial Properties: At creation, the DID can be associated with core attributes.

- **Runtime ID Adaptation & Ephemeral Identities:**

Agents may need to operate under different personas or with limited-scope identities for specific tasks.

- Role-Based/Task-Specific IDs: An agent might present a specific VC that grants it a temporary role or use a derived, short-lived DID.
- Secure Protocol for Assuming Runtime IDs:
  - 1) Request: The agent requests a new role/ephemeral ID/VC.
  - 2) Verification: Issuer verifies primary DID and policies.
  - 3) Issuance: Issuer provides a new (potentially time-bound, scope-limited) VC or ephemeral DID.
  - 4) Usage: Agent uses the new ID/VC for the specific context.
  - 5) Revocation/Expiry: The temporary ID/VC is revoked or expires.

- **ID Update and Revocation:**

- DID Document Updates: Changes to an agent’s capabilities or keys require updating its DID Document.
- VC Revocation: Invalid VCs must be revoked using mechanisms like VC Status Lists.
- DID Deactivation/Revocation: The primary DID can be marked as deactivated if the agent is decommissioned.

This rich, dynamic, and verifiable Agent ID serves as the cornerstone of the proposed Agentic AI IAM framework. The demo SDK for Agent ID is published as open source code at Github [29].

## IV. THE NEW AGENTIC AI IDENTITY AND ACCESS MANAGEMENT FRAMEWORK ARCHITECTURE

To address the multifaceted challenges of managing AI agents in MAS, we propose a comprehensive IAM framework built upon modern cryptographic primitives and a layered architecture designed for dynamic, secure, and interoperable agent interactions.

### *A. Foundational Pillars*

The framework rests on several key technological pillars:

- (A) **Decentralized Identifiers (DIDs) and Verifiable Credentials (VCs):** DIDs [26] provide globally unique, persistent, cryptographically verifiable identifiers controlled by the agent or its controller, enabling self-sovereign identity essential for cross-organizational and decentralized MAS. VCs [27], [28] are digitally signed attestations about an agent, allowing granular and dynamic proof of attributes, capabilities, or authorizations. These technologies are particularly well-suited for representing Non-Human Identities (NHIs), which are widely discussed in the industry [30], [31], providing a standardized approach to managing autonomous agent identities in distributed systems.
- (B) **Zero-Knowledge Proofs (ZKPs):** ZKPs [32] allow an agent to prove a statement’s truth (e.g., possessing a specific VC attribute) without revealing the underlying information, balancing verifiability with privacy. This is crucial for selective disclosure and proving policy compliance without exposing sensitive internal states.
- (C) **Agent Naming and Discovery Service (ANS):** An ANS, inspired by DNS but tailored for agents, enables secure and reliable discovery based on capabilities, protocols, providers, and versions, not just names [33]. This could use a naming structure like `protocol://AgentFunction.CapabilityDomain.Provider.Version[.protocolExtension]` and resolve to DIDs, with entries secured by PKI or linked to verifiable claims.

### *B. Core Architectural Layers*

- 1) **Layer 1: Identity & Credential Management Layer:** Responsible for creating, issuing, storing, and managing the lifecycle of Agent DIDs and VCs.
  - DID Registries/Methods: Systems anchoring DIDs and their DID Documents (e.g., public/permissioned DLTs, did:web, an “Agent ID Provider Network”).
  - VC Issuers and Verifiers: Trusted entities issuing and checking VCs.
  - Agent Wallets/Secure Storage: Secure agent-side storage for private keys and VCs.
  - Key Management Services: For key generation, rotation, and revocation.
- 2) **Layer 2: Agent Discovery and Trust Establishment Layer:** Enables agents to find each other and establish trust.
  - ANS Resolution Mechanisms: Services implementing the ANS for capability-based discovery.
  - DID Resolvers: Standard components for retrieving DID Documents.
  - Reputation Systems: DID-anchored systems for sharing reputation scores.
  - Trust Frameworks: Policies defining how trust is evaluated (e.g., trusted VC issuers).
- 3) **Layer 3: Dynamic Access Control Layer:** Makes fine-grained, context-aware authorization decisions.
  - Policy Decision Point (PDP): Evaluates access requests against policies using agent ID (DID, VCs), resource

attributes, action, and context [34].

- Policy Administration Point (PAP): Where policies (e.g., in Rego/OPA [35]) are defined [34].
- Policy Information Point (PIP): Gathers attributes for the PDP [34].
- Access Control Mechanisms: ABAC, PBAC, and JIT access using temporary, scoped VCs.

4) **Layer 4: Unified Global Session Management & Policy Enforcement Layer:** A critical innovation for consistent, real-time establishment, tracking, management, and enforcement of IAM policies, including global logout and session invalidation, across heterogeneous agent communication protocols.

- Cross-Protocol Session Authority (SA): Logically centralized component for global session oversight, policy distribution, orchestrating global logout, and state change propagation.
- Adapter Enforcement Middleware (AEM): Lightweight plugins injected into Protocol Adapters, hooking into session initiation, subscribing to SA updates (via SSS), intercepting requests, and enforcing decisions locally, including terminating local sessions on global logout.
- Enhanced Protocol Adapters: Gateways understanding specific agent protocols, integrated with AEM for authentication, authorization, and local session management linked to global contexts.
- Session State Synchronizer (SSS): Highly available, low-latency distributed data store maintaining a real-time ledger of active global agent session contexts, their mappings to protocol-specific sessions, and current validated capabilities/status. It's the primary source of truth for AEMs regarding session validity.

*Flow Example: Global Logout for Agent Alpha*

- Global logout for AgentAlpha\_DID reaches SA.
- SA updates SSS: marks GlobalSessionID\_123 (for AgentAlpha\_DID) as "terminated".
- SA may push notifications to relevant AEMs.
- AEM for A2A adapter, on SSS check (or push), sees termination, invalidates local A2A session.
- Similar for MCP adapter's AEM. Further requests from Agent Alpha are blocked.

### C. Applying Zero Trust Principles

The framework embodies Zero Trust [36]–[38]:

- **Explicit Verification:** Always verify agent identity (DID, VCs) and authorization.
- **Least Privilege Access:** Grant minimum necessary permissions, ideally via JIT VCs.
- **Assume Breach:** Design for compromise; rapid revocation via the Unified Enforcement Layer is key.
- **Micro-segmentation:** Granular agent DIDs support network/application micro-segmentation.
- **Data-Centric Security:** Policies tied to data sensitivity and agent capabilities.

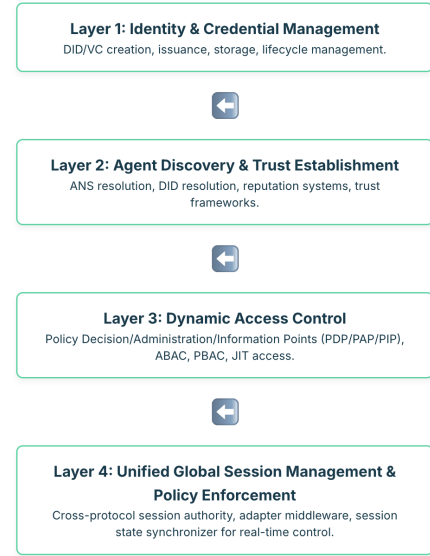


Fig. 1. Core Architecture and its layers

## V. AGENT IDS IN THE IAM PROCESS

This section provides an in-depth exploration of how these constructs enable robust fine-grained access control, ensure secure and non-reputable logging, facilitate effective real-time monitoring and anomaly detection, and empower agile, targeted incident response. A critical enabler for many of these use cases is the Agent Name Service (ANS by [33]), which provides a secure and capability-aware mechanism for agents to discover each other before interaction. We will illustrate conceptual design patterns, including sample interactions involving emerging agent communication protocols like Google's Agent-to-Agent (A2A) protocol [39] and Anthropic's Model Context Protocol (MCP) [15], [16], [40], demonstrating the framework's adaptability and practical utility in complex Multi-Agent Systems (MAS) [41].

### A. Fine-Grained Access Control in Action

Effective access control in MAS must move beyond static roles to embrace dynamic, attribute-based (ABAC), and policy-driven methodologies. The journey often begins with an agent needing to discover another agent or service capable of fulfilling a specific need. This is where the ANS plays a pivotal role, integrated with DIDs and VCs for subsequent secure interaction and authorization.

*Deep Dive into Dynamic Authorization Decisions, Prefaced by ANS Discovery:* Consider TaskOrchestratorAgent (did:com:enterprise:agent:orchestrator:alpha-001) which needs to delegate a financial data analysis task. Its first step is to find a suitable agent. It queries the Agent Name Service (ANS) for an agent that matches certain criteria.

1. *ANS Discovery Phase:* TaskOrchestratorAgent constructs an ANS query. The ANS is designed for capability-aware resolution, using a structured naming convention such as: [Protocol://AgentID.agentCapability.Provider.vVersion.Extension](#).

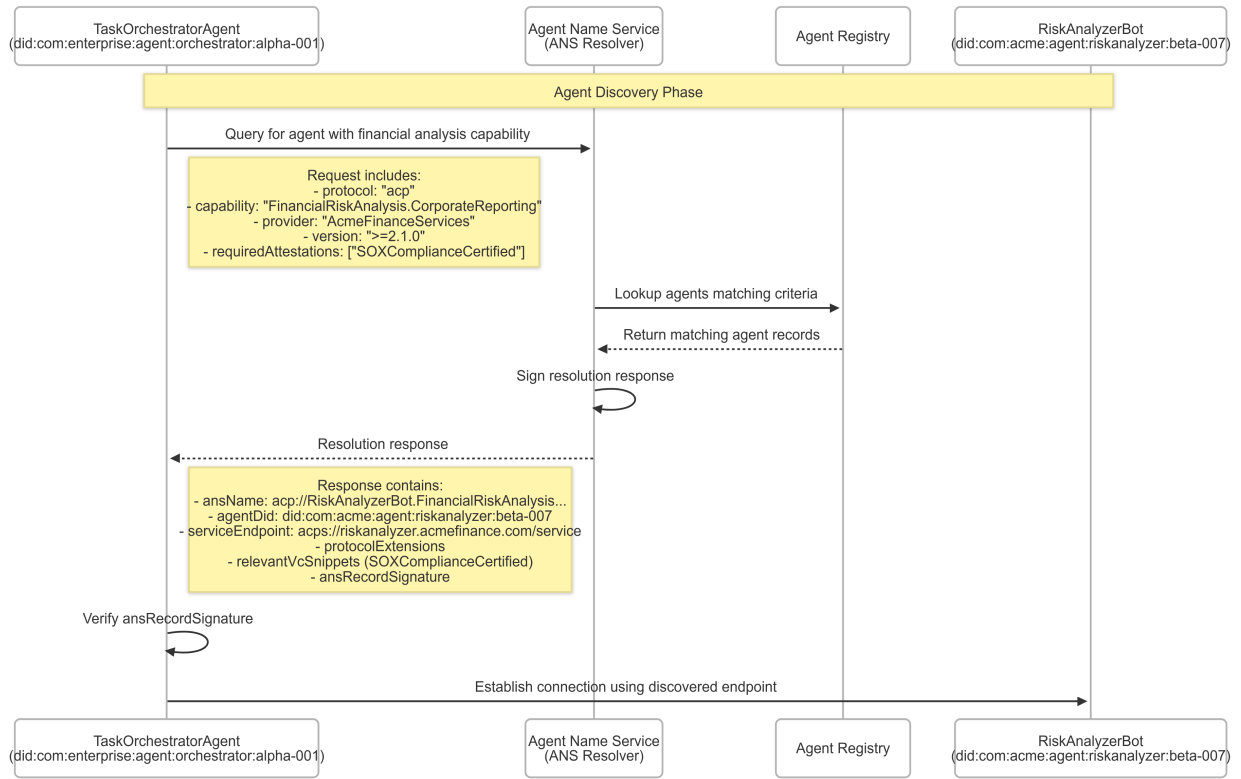


Fig. 2. Agent discovery process using the Agent Name Service (ANS)/

Conceptual ANS Query (e.g., via a secure API call to an ANS resolver):

```

1 // Request to ANS Resolver
2 {
3   "requestType": "resolveAgentByCapability",
4   "desiredProtocol": "acp",
5   "requiredCapability":
6     "FinancialRiskAnalysis.CorporateReporting",
7   "preferredProvider": "AcmeFinanceServices",
8   "versionRange": ">=2.1.0 <3.0.0",
9   "requiredAttestations": [
10     { "vcType": "SOXComplianceCertified" }
11   ]
12 }
  
```

Listing 1. Conceptual ANS Query

The ANS resolver (itself a secure, trusted component of the IAM framework, potentially with its own DID and verifiable responses) queries its Agent Registry [42]. The Agent Registry stores information about registered agents, including their ANSNames, DIDs, PKI certificates (if using a PKI-centric ANS as described in your paper), and protocolExtensions detailing their capabilities and associated VCs.

Conceptual ANS Resolution Response:

```

1 // Response from ANS Resolver
2 {
3   "resolutionStatus": "success",
4   "resolvedAgents": [
5     {
6       "ansName":
  
```

```

    "acp://RiskAnalyzerBot.FinancialRiskAnalysis"
    + ".AcmeFinanceServices.v2.1.3.prod",
    "agentDid":
    "did:com:acme:agent:riskanalyzer:beta-007",
    "serviceEndpoint":
    "acps://riskanalyzer.acmefinance.com/service",
    "protocolExtensions": {
    "acp": { "supportedMessagePatterns":
    ["request-response",
    ↪ "publish-subscribe" ] }
    },
    "relevantVcSnippets": [
    { "type": "SOXComplianceCertified",
    "issuer": "did:com:acme:audit:sox-issuer",
    "issueDate": "2025-01-15" }
    ],
    "ansRecordSignature": "..."
  }
  // Potentially other matching agents
}
  
```

Listing 2. Conceptual ANS Resolution Response

TaskOrchestratorAgent verifies the ansRecordSignature. It now has the DID of a candidate: RiskAnalyzerBot (did:com:acme:agent:riskanalyzer:beta-007).

**2. Interaction and Dynamic Authorization:** TaskOrchestratorAgent now initiates communication with RiskAnalyzerBot (e.g., via ACP). As part of establishing this secure channel or with its first request, RiskAnalyzerBot needs to access InternalDB-SalesFigures and ExternalAPI-MarketSentiment.



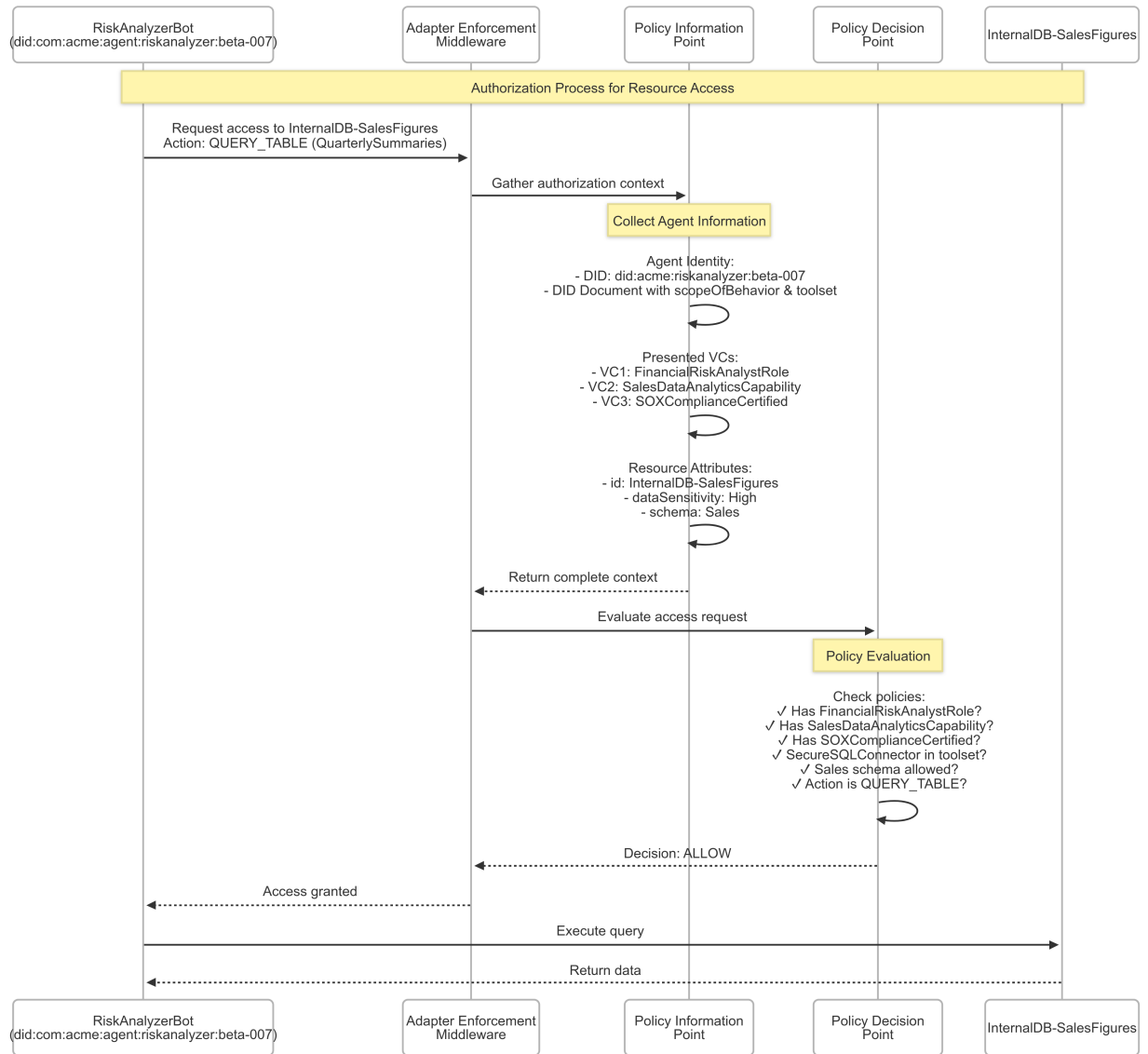


Fig. 3. Fine-grained access control enforcement when RiskAnalyzerBot requests access to sensitive financial data.

The request from RiskAnalyzerBot (let's call it `did:acme:riskanalyzer:beta-007`) to access InternalDB-SalesFigures is intercepted by the Adapter Enforcement Middleware (AEM, see section IV). The AEM/PIP gathers:

- Agent Identity: RiskAnalyzerBot's DID: `did:acme:riskanalyzer:beta-007`. Its resolved DID Document might state: `scopeOfBehavior: "Perform financial risk analysis based on sales and market data." toolset: {"toolName": "SecureSQLConnector", "targetSchemas": ["Sales", "Projections"]}`.
- Presented VCs (obtained during its registration or dynamically):
  - VC1 (Role): { "type": "FinancialRiskAnalystRole", "issuer": "did:com:acme:hr", ... }
  - VC2 (Capability): { "type": "SalesDataAnalyticsCapa-

- VC3 (SOX Compliance - discovered via ANS): { "type": "SOXComplianceCertified", "issuer": "did:com:acme:audit:sox-issuer", ... }

- Resource Attributes: `id: InternalDB-SalesFigures, dataSensitivity: High.`
- Action: `QUERY_TABLE (QuarterlySummaries).`
- Context: `requestTime, sourceIpSegment.`

The PDP evaluates this against policies. For example:

```

1 package acme.data_access
2
3 default allow = false
4
5 # Allow access if agent has correct role,
6 # capability VCs, SOX compliance,
7 # and the requested action is within its
8 # declared toolset capabilities for the resource.
  
```

```

9 allow {
10   input.agent.vcs[_].credentialSubject.role ==
11     ↪ "FinancialRiskAnalystRole"
12   input.agent.vcs[_].credentialSubject.capability
13     ↪ == "SalesDataAnalyticsCapability"
14   input.agent.vcs[_].type[_] ==
15     ↪ "SOXComplianceCertified"
16
17   # Verify toolset from resolved DID Document
18   # (assuming toolset populated by PIP)
19   some tool_idx
20   allowed_tool :=
21     ↪ input.agent.did_document.service[_].
22       serviceEndpoint.toolset[tool_idx]
23   allowed_tool.toolName == "SecureSQLConnector"
24   input.resource.schema IN
25     ↪ allowed_tool.targetSchemas # e.g., "Sales"
26   input.resource.id == "InternalDB-SalesFigures"
27   input.action == "QUERY_TABLE"
28   input.resource.table == "QuarterlySummaries" #
29     ↪ More granular check
30 }

```

Listing 3. Example Rego Policy for Data Access

The ANS discovery step ensures that TaskOrchestratorAgent doesn't just find an agent, but finds one that verifiably claims relevant capabilities and compliance (like SOXCompliance-Certified) before even attempting interaction. The subsequent authorization then re-verifies these claims (via presented VCs) and checks against more granular policies for resource access. This two-step process (secure discovery then secure, fine-grained authorization) is crucial for building trust and efficiency in large MAS. The DID is the consistent thread linking the discovered entity in ANS to the entity being authorized.

*Just-In-Time (JIT) Access, Enhanced by ANS for Tool Discovery:* Imagine DataProcessingAgent-Temp77 (did:ephemeral:task-xyz:agent-77) is a short-lived agent spawned by WorkflowEngine to perform a specific data transformation. It needs temporary access to a specialized DataTransformationTool-Q.

ANS for Tool Discovery: WorkflowEngine (or DataProcessingAgent-Temp77 itself if it has this capability) first queries the ANS to discover a suitable and currently available instance of DataTransformationTool-Q. ANS Query:

```

1 {
2   "requestType": "resolveAgentByNameAndCapability",
3   "ansNamePattern":
4     ↪ "mcp://DataTransformationTool-Q.*"
5     + ".AcmeTools.v1.*.internal",
6   "requiredCapability":
7     ↪ "VectorEmbeddings.HighDimReduction",
8   "availabilityRequirement": "online_accepting_jobs"
9 }

```

Listing 4. ANS Query for Tool Discovery

The ANS returns the DID of an available instance, e.g., did:com:acmetools:mcp:tool:transformQ:instance03.

JIT VC Issuance via MCP Context (Conceptual): WorkflowEngine (acting as a trusted issuer for this context) issues a JIT VC to DataProcessingAgent-Temp77:

```

1 {
2   "type": ["VerifiableCredential",
3     ↪ "MCPToolAccessPass"],

```

```

3   "issuer": "did:com:acme:workflow:engine-issuer",
4   "validFrom": "2025-10-02T14:30:00Z",
5   "validUntil": "2025-10-02T14:45:00Z", // Valid
6     ↪ for 15 mins
7   "credentialSubject": {
8     "id": "did:ephemeral:task-xyz:agent-77",
9     "authorizedToolDID": "did:com:acmetools:mcp:"
10      + "tool:transformQ:instance03",
11     "allowedActions": ["executeTransform"],
12     "inputDataHandle": "blob://temp-input-xyz",
13     "outputDataHandle": "blob://temp-output-xyz",
14     "jobId": "job-ephemeral-77a"
15   }

```

Listing 5. JIT Verifiable Credential for MCP Tool Access

MCP Tool Invocation with JIT VC: DataProcessingAgent-Temp77 invokes DataTransformationTool-Q (whose MCP endpoint was found via ANS then DID resolution). It presents this JIT VC within the MCP call.

Conceptual MCP Call (e.g., using gRPC or HTTP, carrying VC in metadata/headers): Let's assume MCP uses gRPC and metadata for auth as customized transport.

```

1 // Conceptual .proto definition for an MCP tool call
2 service TransformationTool {
3   rpc ExecuteTransform(TransformRequest) returns
4     ↪ (TransformResponse);
5 }
6 message TransformRequest {
7   string job_id = 1;
8   string input_data_reference = 2; //
9     ↪ "blob://temp-input-xyz"
10   map<string, string> transform_parameters = 3;
11 }
12 // Client-side pseudocode for
13   ↪ DataProcessingAgent-Temp77:
14 # Assume 'mcp_tool_stub' is the gRPC stub for
15   ↪ DataTransformationTool-Q
16 # Assume 'jit_vc_jwt' is the JIT VC serialized as a
17   ↪ JWT
18 metadata = [
19   ('x-agent-did',
20     ↪ 'did:ephemeral:task-xyz:agent-77'),
21   ('authorization-vc', jit_vc_jwt)
22 ] # gRPC metadata
23 request_payload = TransformRequest(
24   job_id="job-ephemeral-77a",
25   input_data_reference="blob://temp-input-xyz",
26   transform_parameters={"algorithm": "PCA",
27     ↪ "dimensions": 128})
28 try:
29   response = mcp_tool_stub.ExecuteTransform
30     (request_payload, metadata=metadata)
31   # Process response and write to
32     ↪ "blob://temp-output-xyz"
33 except grpc.RpcError as e:
34   # Handle authorization failure or tool error
35   log(f"MCP tool call failed: {e.details()}")

```

Listing 6. Conceptual MCP Tool Call with JIT VC

Verification at MCP Tool's AEM: The AEM for DataTransformationTool-Q extracts and verifies the DID and jit\_vc\_jwt. The PDP checks if did:ephemeral:task-xyz:agent-77 is authorized by this specific VC to call this tool instance (did:com:acmetools:mcp:tool:transformQ:instance03) for

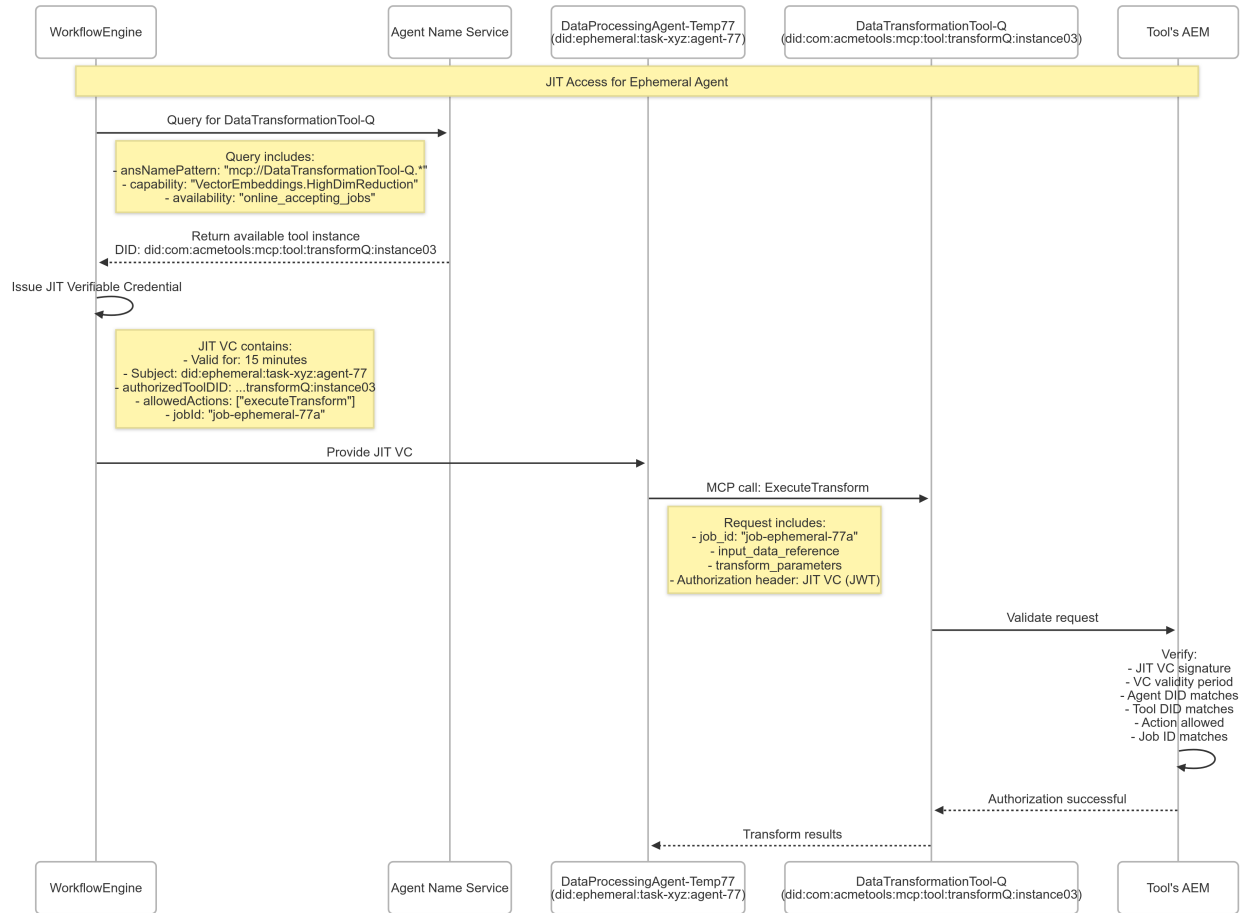


Fig. 4. Ephemeral agent authorization using Just-In-Time (JIT) Verifiable Credentials for Model Context Protocol (MCP) tool access.

executeTransform with the given jobId and data handles, and if the VC is within its validity period.

ANS helps find the right instance of a potentially multi-instance MCP tool. The JIT VC then provides extremely narrow, time-bound permission for that specific job and data, dramatically reducing risk compared to the ephemeral agent having broader, longer-lived credentials for a generic tool type.

**Capability-Driven Authorization with A2A Protocol:** AlertingAgent-SystemX (did:com:sysX:a2a:alerter:main:v1) needs to send a critical security alert to a SOCDashboardAgent-PlatformY (did:com:platY:a2a:socdash:primary:v2).

**ANS Discovery:** AlertingAgent-SystemX resolves [a2a://SOCDashboardAgent.SecurityAlertIngestion.PlatformY.v2.critical](#) via ANS to find the DID and A2A endpoint of SOCDashboardAgent-PlatformY. The ANS response might also indicate that the SOC agent requires alerts to be signed with a key whose DID is on an approved list.

**A2A Message Construction with IAM Context:** AlertingAgent-SystemX holds a VC: {"type": "CriticalAlert-SourceCredential", "issuer": "did:com:sysX:security-authority", "credentialSubject": {"id": "did:com:sysX:a2a:alerter:main:v1", "authorizedAlertTypes": ["SECURITY\_CRITICAL",

"SYSTEM\_DOWN"]}}.

**Conceptual A2A Message from AlertingAgent-SystemX (JSON-like payload for an A2A message):**

```

1 {
2   "a2aHeader": {
3     "messageId": "msg-uuid-9876",
4     "senderId": "did:com:sysX:a2a:alerter:main:v1",
5     "recipientId":
6       ↪ "did:com:platY:a2a:socdash:primary:v2",
7     "protocolVersion": "A2A/1.0"
8   },
9   "iamExtension": {
10    "verifiablePresentation": [ /* JWT of
11      ↪ CriticalAlertSourceCredential */ ],
12    "messageSignature": {
13      "keyId":
14        ↪ "did:com:sysX:a2a:alerter:main:v1#key-1",
15      "algorithm": "EdDSA",
16      "signatureValue": "..."
17    }
18  },
19  "payload": {
20    "alertType": "SECURITY_CRITICAL",
21    "sourceSystem": "SystemX_Firewall_Cluster",
22    "details": "Multiple intrusion attempts detected
23      ↪ from IP range Z.Z.Z.Z",
24    "severity": 5,
25    "timestamp": "2025-10-02T15:00:10Z"
26  }
27 }
  
```

```
22 }  
23 }
```

Listing 7. Conceptual A2A Message with IAM Context

Many emerging A2A protocols are defining ways to carry security contexts, often leveraging JWTs or similar token formats within their headers or as part of the message envelope. The `iamExtension` is a way our framework's specific needs (DID, VP) can be mapped.

Processing at SOCDashboardAgent-PlatformY's AEM:

- AEM verifies `messageSignature` using the public key from `did:com:sysX:a2a:alerter:main:v1#key-1` (resolved via DID document).
- AEM verifies the `verifiablePresentation` containing the `CriticalAlertSourceCredential`.
- PDP checks policies like: "Accept SECURITY\_CRITICAL alert IF sender DID holds valid `CriticalAlertSourceCredential` AND the alert's declared `sourceSystem` is within the scope covered by that credential."

The ANS ensures `AlertingAgent-SystemX` reliably finds the authentic `SOCDashboardAgent-PlatformY` (not an imposter). The VC presented proves the sender is authorized to issue critical alerts, and the message signature ensures integrity and non-repudiation for the alert content. This provides much stronger guarantees than simple IP whitelisting or pre-shared API keys between agents for A2A communication [43].

The use of ANS for initial discovery, followed by DID-based authentication and VC-based authorization at the point of interaction, forms a robust sequence for secure and fine-grained access control in diverse MAS scenarios.

The Identity and Access Management (IAM) framework for multi-agent systems operates through a sophisticated four-phase lifecycle that ensures secure agent discovery, authentication, and authorization across diverse protocol environments. As illustrated in Figure 2, the process begins with capability-aware agent discovery through the Agent Name Service (ANS), where requesting agents query for specific capabilities, compliance requirements, and protocol preferences, receiving cryptographically signed responses containing target agent DIDs, service endpoints, and relevant attestations such as SOX compliance certifications. Once agents establish contact, the framework employs dynamic, attribute-based access control as demonstrated in Figure 3, where the Adapter Enforcement Middleware (AEM) coordinates with Policy Information Points (PIP) to gather comprehensive agent context including Decentralized Identifiers (DIDs), Verifiable Credentials (VCs), and resource attributes, before Policy Decision Points (PDP) evaluate fine-grained authorization policies that consider roles, capabilities, toolset permissions, and data sensitivity levels. For ephemeral or temporary agents, the framework supports Just-In-Time (JIT) credential issuance as shown in Figure 4, where workflow engines discover available tools via ANS and issue time-limited Verifiable Credentials with narrow, job-specific permissions that enable secure Model Context Protocol (MCP) interactions while minimizing credential exposure and

attack surfaces. Finally, the framework facilitates secure inter-agent communication through emerging protocols like Google's Agent-to-Agent (A2A) protocol, as depicted in Figure 5, where agents exchange cryptographically signed messages containing verifiable presentations that prove authorization for specific communication types, ensuring non-repudiation and enabling real-time security alerting between heterogeneous agent platforms while maintaining end-to-end trust and accountability throughout the multi-agent ecosystem.

## B. Secure Logging, Auditing, and Non-Repudiation

In systems where autonomous agents perform significant actions, establishing a clear, trustworthy, and irrefutable record of events is paramount. This section delves into how the proposed IAM framework, leveraging rich Agent IDs (DIDs and VCs) and the Agent Name Service (ANS) for discoverable context, transforms logging into a critical component of system integrity, accountability, and auditability.

*Immutable Agent Identifiers (DIDs) as the Linchpin of Audit Logs:* Every significant action initiated or participated in by an agent MUST be logged with its unique, persistent Decentralized Identifier (DID) as the primary subject identifier. This creates an unambiguous, globally unique, and cryptographically verifiable link to the specific agent instance responsible for any given event.

*Enhanced Log Granularity with DID and VC Context:* Beyond simply logging the agent's DID, comprehensive logs should capture:

- **Precise Timestamp:** Synchronized across the MAS to ensure correct event sequencing.
- **Agent DID and ANSName:** Logging both the DID (for cryptographic verifiability) and the resolved ANSName (e.g., [acp://RiskAnalyzerBot.FinancialRiskAnalysis.AcmeFinanceServices.v2.1.3.prod](#)) provides human-readable context about the agent's role and origin.
- **Target Resource(s) DIDs/ANSNames:** If the interaction target is another agent or a resource registered in ANS, its DID and ANSName should also be logged.
- **Input Parameters/Data Hashes:** Hashing critical inputs helps reconstruct the context of an agent's decision without necessarily storing sensitive raw data in logs.
- **Specific Verifiable Credentials (VCs) Presented:** The unique identifiers (e.g., `id` or `transaction_id`) of all VCs presented by the agent to authorize that specific action. For example, logging `vc:jwt:uri:issuer-finance-bob:task-q3report2025-instance-002` allows an auditor to later retrieve and verify this exact VC.
- **DIDs and ANSNames of Collaborating Agents:** In multi-agent tasks, the DIDs/ANSNames of all significant contributing agents should be logged to trace collaborative decision-making.
- **Outcome and Policy Reference:** The result of the action and a reference to the specific policy version (e.g., `ACME_Finance_Policy_v3.2.1_Rule7`) that permitted it.

Example Enriched Log Entry incorporating ANSNames:



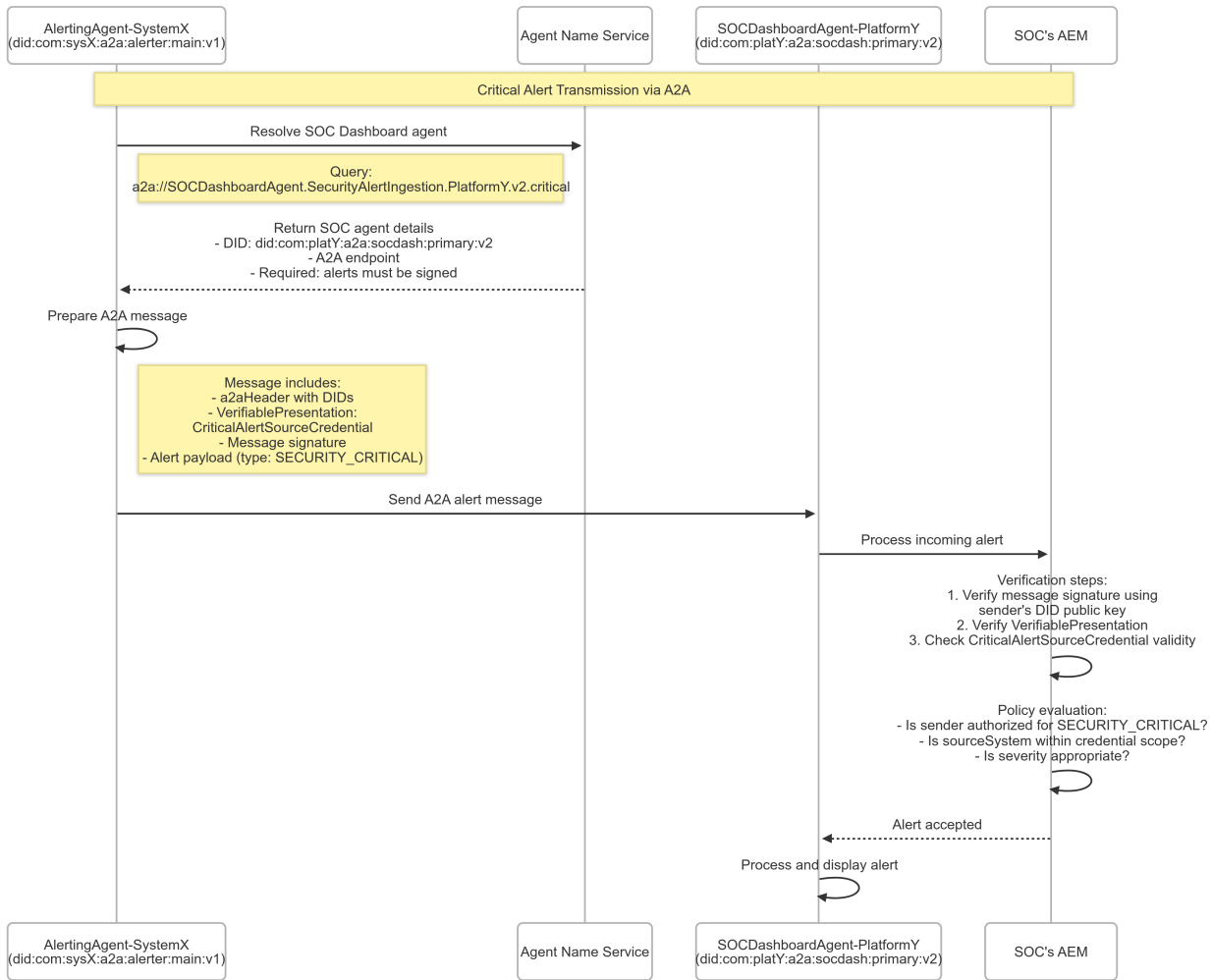


Fig. 5. Secure agent-to-agent communication using Google's A2A protocol for critical security alerts.

```

1 {
2   "eventId": "evt_20251002T110530Z_A789F123",
3   "timestamp": "2025-10-02T11:05:30.123Z",
4   "initiatingSystem":
5     ↳ "WorkflowOrchestratorInternal",
6   "agentDid":
7     ↳ "did:com:acme:agent:riskanalyzer:beta-007",
8   "agentAnsName":
9     "acp://RiskAnalyzerBot.FinancialRiskAnalysis"
10    + ".AcmeFinanceServices.v2.1.3.prod",
11   "actionPerformed": "ExecuteSecureSQLQuery",
12   "targetResourceDid":
13     "did:com:acme:resource:db:InternalDB-SalesFigures",
14   "targetResourceAnsName":
15     "db://InternalDBSales.FinancialData"
16     + ".AcmeInternal.v1.prod",
17   "inputParametersHash": "sha256-c4d5e6f...",
18   "presentedVcIds": [
19     "vc:jwt:uri:acme-hr:role-finanalystL2-inst-001",
20     "vc:jwt:uri:acme-audit:sox-compliance-inst-003"
21   ],
22   "decisionPolicyId":
23     "ACME_DataAccess_Policy_v1.7_Rule12b",
24   "collaborationContext": {
25     "triggeringAgentDid":
26       "did:com:enterprise:agent:orchestrator:alpha-001",
27     "triggeringAgentAnsName":
28       ↳ "acp://TaskOrchestrator.CoreBusinessLogic"
29       + ".AcmeEnterprise.v1.0.main",
30     "taskId": "task_QuarterlyRiskAssessment_2025Q3"
31   },
32   "outcome": { "status": "Success", "rowsAffected":
33     ↳ 0, "dataRetrievedHash": "sha256-g7h8i9j..."
34     ↳ },
35   "logEntrySignature": "..."
36 }

```

Listing 8. Enriched Log Entry with ANSNames

Logging ANSNames alongside DIDs makes logs instantly more interpretable for human auditors. The cryptographic link via DIDs ensures the identifier is not just a mutable string. The logged VCs provide the exact authorization context for the action, making audits far more precise.

**Cryptographic Non-Repudiation of Agent Actions via DID Signatures:** To achieve strong non-repudiation, critical agent actions or the data they produce can be digitally signed by the agent using the private key associated with its DID. This

is particularly important for actions with financial, legal, or safety implications.

Scenario (A2A Context): OrderPlacementAgent (did:com:retail:a2a:orderbot:v1.0, ANSName [a2a://OrderPlacement.RetailTransactions.MegaCorp.v1.0.live](#)) submits a purchase order to SupplierFulfilmentAgent (did:com:supplierX:a2a:fulfill:v2.1, ANSName [a2a://Fulfilment.SupplyChain.SupplierX.v2.1.prod](#)), which was discovered via ANS query for "SupplyChain.OrderFulfilment.SupplierX".

A2A Message with Signed Payload and DID Context: The OrderPlacementAgent constructs an A2A message. The core business payload (the order details) is signed.

```

1 // A2A Message (Conceptual JSON representation)
2 {
3   "a2aHeader": {
4     "messageId": "order-uuid-554433",
5     "senderId": "did:com:retail:a2a:orderbot:v1.0",
6     "recipientId":
7       ↪ "did:com:supplierX:a2a:fulfill:v2.1",
8     "protocolVersion": "A2A/1.0",
9     "timestamp": "2025-10-02T16:30:00Z"
10  },
11  "iamExtension": {
12    "verifiablePresentation": [ /* Optional: JWT of
13      ↪ a relevant VC */ ]
14  },
15  "payload": {
16    "orderId": "PO-2025-10-778",
17    "items": [ { "sku": "XYZ123", "quantity": 100 },
18      ↪ { "sku": "ABC789", "quantity": 50 } ],
19    "shippingAddress": "123 Main St, Anytown",
20    "totalAmount": 12500.75,
21    "currency": "USD"
22  },
23  "payloadSignature": {
24    "keyId":
25      "did:com:retail:a2a:"
26      + "orderbot:v1.0#key-transact",
27    "algorithm": "EdDSA",
28    "signatureValue": "..."
29  }
30 }

```

Listing 9. A2A Message with Signed Payload

Verification and Logging by SupplierFulfilmentAgent:

- The AEM at SupplierFulfilmentAgent's side first authenticates the sender via its DID and any presented VCs (as per Section V-A).
- It then specifically verifies the payloadSignature using the public key did:com:retail:a2a:orderbot:v1.0#key-transact (obtained by resolving the sender's DID).
- SupplierFulfilmentAgent's log entry for receiving this order would include: its own DID/ANSName, the sender's DID/ANSName, the order ID, a hash of the received payload, and the payloadSignature object. This creates a verifiable record that OrderPlacementAgent indeed sent that specific order. The initial discovery via ANS ensures the order is sent to a legitimate fulfilment agent. The DID-based signature on the payload provides strong non-repudiation for the order's content, traceable to a specific, verifiable agent identity. Traditional EDI or API calls often rely on weaker authentication or channel security alone.

*Verifiable Provenance Chains in MCP Tool Interactions:* When an LLM-based agent uses an MCP tool, understanding the full chain—from user prompt to LLM, to MCP tool call, to tool result, back to LLM, and then to the user—is vital for auditing and debugging.

Scenario: A user asks ResearchLLM-Agent (did:com:ai-lab:mcp:researcher:zeta:v3.1, ANSName [mcp://Researcher.ScientificQuery.AILab.v3.1.experimental](#)) a complex question requiring a database lookup via an MCP tool, SemanticSearchTool (did:com:datastore:mcp:tool:semsearch:v1.0, ANSName [mcp://SemanticSearch.KnowledgeBase.DataCorp.v1.0.main](#)). ResearchLLM-Agent discovers SemanticSearchTool via an ANS query specifying the "KnowledgeBase.SemanticSearch" capability.

MCP Interaction Logging with DIDs and VCs:

- User Interaction Log: User prompt, timestamp, and ResearchLLM-Agent's DID/ANSName.
- ResearchLLM-Agent Internal Log (or trace):
  - Decision to use SemanticSearchTool.
  - Query sent to ANS for SemanticSearchTool.
  - Resolved DID/ANSName for SemanticSearchTool.
  - The MCP call it constructs to SemanticSearchTool, including:
    - \* Its own DID as the caller.
    - \* The JIT VC it obtained/presented for this tool use (e.g., vc:jwt:...:mcp-tool-access-zeta-job778).
    - \* The parameters sent to the tool.
  - This entire MCP call could be signed by ResearchLLM-Agent.
- SemanticSearchTool (MCP Tool) Log:
  - Its own DID/ANSName.
  - Receiving the MCP call from ResearchLLM-Agent (DID/ANSName logged).
  - The presented JIT VC ID.
  - Verification status of the caller's DID and VC.
  - Parameters received.
  - Actions it took (e.g., database queries it made internally).
  - The result it returned to ResearchLLM-Agent.
  - This log entry or the response payload could be signed by SemanticSearchTool.
- ResearchLLM-Agent Internal Log (continued):
  - Response received from SemanticSearchTool (potentially with signature verification).
  - How it processed the tool's output.
  - The final answer generated for the user (this answer could also be signed).

This chained logging, where each step is linked by verifiable DIDs/ANSNames and specific VCs or signed messages, creates a rich, end-to-end auditable provenance trail. If the final answer is wrong, auditors can trace back: Was it the LLM's reasoning, the MCP tool's execution, the data the tool accessed, or the initial ANS discovery that pointed to an incorrect tool version? This detailed, verifiable chain is crucial for explainability and

accountability in complex agentic workflows involving external tools.

*Privacy-Preserving Audits of IAM Policies with ZKPs:* Organizations may need to prove to external auditors or regulators that their Agentic AI IAM policies are being correctly enforced, without revealing the proprietary details of all policies or all agent interactions.

Scenario: An auditor wants to verify that access to resources tagged PII\_Strict is only ever granted if an agent presents a valid VC of type PII\_AccessLevel3\_Certified and the request originates from an approved network segment. Mechanism:

- The IAM system's Policy Decision Point (PDP) logs all its decisions, including the agent DID, resource, action, presented VCs (or their hashes), contextual attributes, and the allow/deny outcome. These logs themselves could be cryptographically committed to (e.g., a hash chain).
- The organization can run a process that analyzes these logs and generates a ZKP. This ZKP would prove a statement like: "For all access requests to resources tagged PII\_Strict within the last audit period that resulted in an 'allow' decision, the requesting agent's presented credentials included a valid (non-revoked, correctly signed) PII\_AccessLevel3\_Certified VC from an approved issuer, AND the source network attribute was in the set {'segA', 'segB'}."
- This ZKP is generated without revealing the specific agent DIDs, resource DIDs, exact times, or other details of the individual access events.
- The auditor receives and verifies this ZKP, along with information about the approved VC issuers and network segments, providing strong assurance of policy enforcement without seeing the raw, potentially sensitive log data. This enables "compliance as code" verification with privacy. It allows organizations to demonstrate adherence to internal or external IAM rules without exposing the minutiae of every transaction, which is a common challenge in traditional audit processes that often require extensive (and risky) data sharing.

By deeply integrating verifiable Agent IDs (DIDs/VCs), secure discovery via ANS, and cryptographic techniques like digital signatures and ZKPs into the logging and auditing process, our framework aims to create a system where agent actions are not just recorded, but are verifiably attributable, contextualized, and, where necessary, proven compliant in a privacy-respecting manner. This robust auditability is fundamental to building and maintaining trust in complex and autonomous MAS.

### C. Real-time Monitoring and Anomaly Detection

Effective IAM extends beyond static policy enforcement to encompass continuous, real-time oversight of agent activities. The rich, verifiable Agent IDs (DIDs and VCs), coupled with the contextual information available through Agent Name Service (ANS) resolutions, provide the foundation for a far more sophisticated and proactive monitoring and anomaly detection capability than achievable with traditional, opaque identifiers. This allows security systems to not only identify

what is happening but also understand if it aligns with an agent's intended and attested purpose and capabilities.

*Establishing Rich Behavioral Baselines Anchored to Verifiable Identities (DIDs and ANSNames):* Modern monitoring can move beyond tracking simple metrics like CPU usage per IP address. The proposed framework allows for the creation of multifaceted behavioral baselines for each unique agent DID and its associated ANSName profiles:

- **Discovered vs. Declared Scope of Behavior:** The agent's DID Document contains its scopeOfBehavior (e.g., "customer\_support\_query\_resolution\_for\_product\_X"). ANS registration might also include a primary capability (e.g., Support.ProductQuery.CustomerFacing.v1). Monitoring systems can compare the agent's actual interactions and data access patterns against this declared and discoverable purpose. Significant deviations trigger alerts.

Scenario: SupportAgentAlpha (did:com:support:agent:alpha01, ANSName [helpdesk://Support.ProductQuery.CustomerFacing.v1.Acme](#)) normally accesses the product knowledge base and customer ticket system. If it suddenly starts making frequent ANS queries for agents with FinancialData.InternalAudit capabilities, or attempts to access database schemas related to payroll, this is a strong anomaly relative to its declared/discovered scope.

- **Authorized Toolset and ANS-Discoverable Service Usage:** The agent's DID Document details its toolset (specific APIs, other agent DIDs/ANSNames it's authorized to interact with). Monitoring systems can track: Actual tool/API calls made. ANS queries made by the agent to discover other services. If the agent attempts to use tools not in its list or interact with DIDs/ANSNames that don't match its typical collaboration patterns or authorized interaction VCs.

Scenario (MCP Context): DataPipelineAgent-ETL (did:com:dataops:agent:etl04, ANSName [mcp://ETL.DataWarehouseLoading.DataOps.v2.nightly](#)) is authorized to use PostgresConnectorTool (an MCP tool discovered via ANS as [mcp://DBConnector.PostgreSQL.InternalTools.v1.stable](#)) and S3StorageTool. If it makes an ANS query for [mcp://ExternalAPI.SocialMediaScraping...](#) or attempts to invoke such a tool via MCP, it's a policy violation and an anomaly.

- **VC Presentation Patterns:** Monitoring the types of VCs an agent typically presents for different actions, and the issuers of those VCs. An agent suddenly presenting a VC from a previously unseen or untrusted issuer for a high-privilege operation is suspicious.
- **Communication Graph and Trust Dynamics:** Building a graph of typical agent-to-agent interactions (DID-to-DID or ANSName-to-ANSName) based on historical communication logs. New, unexpected communication links, especially with agents outside the organization or with low reputation scores (if a reputation system is

integrated), can be flagged.

*Scenario:* A fleet of InventoryCheckAgent instances (e.g., `a2a://InventoryCheck.RetailStoreXYZ.Ops.v1.hourly::did:...`) typically only communicate via A2A with a central InventoryMasterAgent (`a2a://InventoryMaster.HeadOffice.Ops.v3.main::did:...`). If one InventoryCheckAgent initiates an A2A connection to an unknown external ANSName/DID, or starts sending unusually large A2A payloads, this is anomalous.

*Advanced Deviation Detection Leveraging Verifiable Claims:* The ability to verify claims presented as VCs in real-time enhances anomaly detection:

- **Scope Creep Beyond VC-Attested Capabilities:** An agent, ResearchSummarizer (`did:...`, ANSName `a2a://Summarization.ScientificLiterature.ResearchGroup.v1`), might hold a VC for "Access\_PubMed\_API\_SummarizationOnly." If it attempts to use the PubMed API's "BulkDownloadAbstracts" function (which its VC does not authorize), the AEM/PDP would block it, and the monitoring system would log this as a significant deviation, as it's attempting an action beyond its attested capability.
- **Anomalous JIT VC Requests:** If an agent frequently requests JIT VCs for tasks outside its typical operational parameters, or if the requested scopes for JIT VCs escalate without justification, this could indicate a compromised agent or a misbehaving workflow.
- **Interaction with Agents Lacking Expected Counter-Attestations:** If SecureDataTransferAgent is only supposed to send data to other agents that can present a "DataRecipient\_EncryptionLevel5\_Compliant" VC, an attempt to send data to an agent (discovered via ANS) that cannot present such a VC would be a flagged anomaly, even if basic network connectivity is possible.

*Dynamic Trust Scoring and Risk-Adaptive IAM Incorporating ANS Context:* The Agent ID (DID) becomes the anchor for a dynamic trust score, influenced by monitoring. ANS context adds another layer.

- **Inputs to Trust Score (with ANS context):** Successful completion of tasks within the agent's ANS-declared capability. Policy violations or anomalous behaviors (as detailed above). Validity and issuer trustworthiness of its VCs. Feedback from other reputable agent DIDs (whose own ANS profiles might indicate their roles/trustworthiness). ANS-related anomalies: Repeatedly querying ANS for unrelated capabilities, attempting to register with a misleading ANSName, or interacting with agents resolved from suspicious ANS domains.
- **Risk-Adaptive Policy Enforcement Example (A2A):** PaymentAgent-Acquirer (`a2a://PaymentProcessing.Acquisition.FinServ.v2.live::did:...`) normally processes transactions. It starts making unusual ANS queries for `a2a://DataAggregation.UserProfiling...` services and receives a few low-severity alerts for attempting to access non-payment related internal APIs. Its trust score,

managed by the IAM system, is lowered. The Session Authority (SA) is notified of the trust score change. The SA updates the Session State Synchronizer (SSS) for this agent's global session, adding a "ReducedTrust" status or dynamically adjusting its permissible capability set. When PaymentAgent-Acquirer next attempts a high-value A2A payment authorization request to PaymentGateway-PSP (`a2a://Gateway.PaymentAuth.PSPGlobal.v4.secure::did:...`), the AEM at the gateway side consults the SSS. Even if the agent presents its usual VCs, the SSS indicates "ReducedTrust." The PDP at the gateway might now enforce a stricter policy: "IF agent\_status == 'ReducedTrust', THEN require\_multi\_factor\_agent\_auth (e.g., a ZKP of a recent controller approval for this transaction type) OR limit\_transaction\_value\_to\_low\_threshold." The A2A transaction might be rejected or queued for additional checks, preventing potential fraud by a slightly misbehaving or partially compromised agent.

The ANS provides discoverable context about an agent's intended role and capabilities. Monitoring deviations from this publicly or organizationally declared purpose, in addition to private policy violations, gives a richer signal for anomaly detection. The trust score becomes more robust as it can factor in the consistency of an agent's behavior with its registered identity profile in ANS.

*D. Agile Incident Response: Precision Targeting, Rapid Containment, and Discoverable Impact*

When a security incident occurs, the ability to respond swiftly, precisely, and comprehensively is critical to minimizing damage. The proposed IAM framework, with its integration of DIDs, VCs, and ANS, provides superior capabilities for incident response.

*Rapid and Unambiguous Identification via DID and ANS Context:* Security alerts from monitoring systems or external threat intelligence will directly reference the compromised or malicious agent's DID and often its ANSName. This removes ambiguity and allows response teams to immediately identify: The specific agent instance involved (via DID). Its declared purpose and owner (via ANSName and resolved DID document). Its attested capabilities and dependencies (via VCs and DID document).

*Example:* An alert "Unusual data exfiltration by `did:com:cloudstorage:agent:backup-beta-721` (ANSName: `a2a://Backup.CriticalDB.AcmeCorp.v1.beta.nightly`)" immediately tells the SOC: It's a specific backup agent instance. It's associated with AcmeCorp's critical database backups. It's a beta version (which might imply higher risk or different oversight).

*Targeted Revocation with Ecosystem-Wide Propagation:* The framework supports granular to broad revocation, propagated efficiently:

- **VC Revocation (Surgical):** If a specific attested capability (e.g., VC:AbilityToModifyUserPermissions) of AdminBot-HR (`did:com:hr:adminbot:003`, ANSName `a2a://UserAdmin.Permissions.HRInternal.v2.prod`) is found to



be exploited due to a bug, that VC is added to a VC Status List. AdminBot-HR might still function for other tasks (e.g., reading user profiles) using its other VCs, but attempts to use the revoked permission VC will fail.

- **DID Deactivation/Revocation (Logical via DID Method or ANS):** If AdminBot-HR's private keys are confirmed stolen, its entire DID (did:com:hr:adminbot:003) is revoked via its DID method. The ANS entry for [a2a://UserAdmin.Permissions.HRInternal.v2.prod](#) would then either resolve to a "revoked" status or be removed/updated by the ANS Registration Authority. Other agents querying ANS for this service will no longer receive the compromised DID.
- **Instantaneous Global Session Invalidation via Unified Enforcement Layer:** This is the most critical response.
  - Trigger: SOC confirms did:com:hr:adminbot:003 is actively malicious.
  - SA Notification: The Session Authority (SA) is notified, specifying the DID.
  - SSS Update: SA updates the Session State Synchronizer (SSS) to mark all global sessions for did:com:hr:adminbot:003 as "TERMINATED\_IMMEDIATE\_SECURITY\_LOCKOUT".
  - AEM Enforcement: All AEMs interacting with or receiving requests from did:com:hr:adminbot:003 (whether via A2A, MCP, or internal ACP/HTTP calls) consult the SSS. They see the "TERMINATED" status and instantly block any new requests and terminate any active local protocol sessions.

*Scenario (MCP Tool in use by AdminBot-HR):* If AdminBot-HR was using an MCP tool like UserProvisioningTool, its active MCP session (managed by the tool's AEM) would be killed. Further MCP calls from AdminBot-HR would be rejected by the AEM before even reaching the tool's logic.

*Scenario (A2A communication):* If AdminBot-HR was sending A2A messages to AuditLogAgent, these A2A messages would be blocked by the AEM on AuditLogAgent's side.

The ANS provides a clear point for signaling revocation at the discovery layer. Even if an attacker has cached an old DID, new discovery attempts for the agent's function would fail or return a revoked status. The SSS ensures that active sessions, regardless of how they were initiated (perhaps post-ANS discovery), are comprehensively terminated.

*Rich Forensic Analysis with Discoverable Context:* Post-incident, the combination of DID-anchored logs, VCs, and ANS information provides unparalleled depth for forensics.

- **Contextualizing Compromise:** If did:com:research:agent:dataminer:gamma-9 is compromised, investigators can not only see its actions (via DID logs) but also: Resolve its ANSName ([science://DataMining.LargeDatasets.ResearchDiv.v0.9.experimental](#)) to understand its expected role and provider context. Examine its DID Document and VCs

to see its intended capabilities and dependencies (e.g., "depends on did:com:lib:math:vectorcalc:v3.2"). This helps to check if a dependency was the root cause. Trace its ANS query history: Was it trying to discover and interact with services outside its normal profile before the compromise? If it interacted with other agents, their DIDs/ANSNames are in the logs, allowing investigators to assess the blast radius and check if those collaborators were also affected or were part of the attack.

- **Identifying Attack Vectors via ANS:** If multiple agents registered under a specific, less reputable Provider in their ANSNames are simultaneously compromised, it might indicate a targeted attack against that provider's agent infrastructure or a vulnerability common to their agents.

ANS data (like provider, capability domain in the name) adds valuable metadata for clustering incidents, identifying patterns, and understanding the potential scope or origin of an attack that might involve multiple agent instances from a similar source or with similar functions.

#### *E. Other Potential Uses Building on Verifiable Agent IDs and Discoverable ANS Profiles*

The synergistic use of detailed, verifiable Agent IDs and a structured Agent Name Service, all managed within a robust IAM framework, naturally extends to enable further advanced functionalities critical for a mature and trustworthy AI ecosystem.

#### *Decentralized Reputation and Trust Brokering with ANS-Contextualized Feedback:*

- Agent DIDs serve as the stable anchors for accumulating reputation scores. When AgentA (e.g., discovered via ANS as [a2a://TaskExecutor.GeneralPurpose.CommunityPool.v1.standard::did:agentA...](#)) completes a task for AgentB, AgentB can issue a reputation VC attesting to AgentA's performance, timeliness, and reliability for that specific task type (derived from AgentA's ANS capability).
- These VCs can be stored by AgentA or published to a decentralized reputation ledger. Future agents querying ANS for "TaskExecutor.GeneralPurpose" might then also be able to query this reputation system (using the resolved DID) for community feedback, prioritizing agents with higher, relevant reputation scores. The ANS capability string itself provides context for the reputation (e.g., good at "GeneralPurpose" tasks).

Code Concept: Agent B issuing a reputation VC for Agent A:

```
1 # Agent B's perspective
2 # from pyld import jsonld # For Verifiable
   ↳ Credentials
3 # from did_sdk import sign_vc # Conceptual SDK
   ↳ function
4
5 agent_A_did = "did:agentA..."
6 agent_A_ans_capability =
   ↳ "TaskExecutor.GeneralPurpose."
7   + "CommunityPool.v1.standard"
8
9 reputation_claim = {
10   "@context":
   ↳ ["https://www.w3.org/2018/credentials/v1",
```

```

11     "https://example.org/reputation/v1"],
12     "type": ["VerifiableCredential",
13             ↪ "ReputationCredential",
14             ↪ "PerformanceReview"],
15     "issuer": "did:agentB...", # Agent B's DID
16     "issuanceDate": "2025-10-03T10:00:00Z",
17     "credentialSubject": {
18         "id": agent_A_did,
19         "ansCapabilityContext":
20             ↪ agent_A_ans_capability,
21         "rating": 5, # Scale of 1-5
22         "comment": "Completed task efficiently and
23             ↪ accurately.",
24         "taskId": "task-uuid-for-context"
25     }
26 }
27 # Agent B signs this claim with its DID key to
28     ↪ create a VC
29 # signed_reputation_vc = sign_vc(reputation_claim,
30     ↪ "did:agentB...", "did:agentB...#key-1")
31 # Agent B might then send this VC to Agent A, or
32     ↪ publish it to a reputation service.

```

Listing 10. Agent B Issuing Reputation VC for Agent A

#### *Automated Billing and Resource Quota Enforcement via ANS-Defined Services:*

- When an agent discovers and uses a commercial service (e.g., a specialized MCP tool like [mcp://AdvancedTranslation.Multilingual.PremiumAPI.v3.commercial::did:tool:translateXYZ...](#)) via ANS, the ANS record itself might point to metadata about pricing models or rate limits associated with that service DID.
- The consuming agent's DID is logged by the commercial tool for every API call. The tool provider's AEM/PDP can enforce quotas (e.g., "Agent did:com:startup:agent:translator007 has a quota of 10M characters/month for did:tool:translateXYZ"). Billing is then accurately attributed to the consuming agent's controller.

#### *Secure Software/Model Supply Chain Attestations Linked to ANS Registrations:*

- When an agent is registered with ANS (e.g., [a2a://ImageRecognition.MedicalScans.RadAI.v2.validated::did:radai:imgrec:002](#)), part of its registration with the ANS Registration Authority (RA) could involve presenting VCs that attest to its supply chain security: A VC for its base foundation model (e.g., "Model-Card\_VC\_for\_RadAI\_BaseVisionModel\_v2"), detailing its training data, bias tests, and safety evaluations. SBOM VCs for its software components. A "ValidatedSecure-Build\_VC" from a trusted CI/CD pipeline.
- The ANS resolver could then optionally return indicators of these attestations (or links to the VCs) along with the agent's DID, allowing discoverers to prioritize agents with verifiable supply chain security.

#### *Dynamic Coalition Formation and Capability Negotiation using ANS for Initial Matching:*

- An EmergencyResponseOrchestratorAgent queries ANS for agents with diverse capabilities like [a2a://DroneSurveillance.DisasterZoneMapping...](#), [mcp://Logistics.ResourceAllocation...](#), and [comms://TemporaryNetwork.MeshDeployment...](#)

illance.DisasterZoneMapping..., mcp://Logistics.ResourceAllocation..., and comms://TemporaryNetwork.MeshDeployment...

- Once candidate DIDs are retrieved, the orchestrator can initiate a negotiation phase (e.g., using FIPA Contract Net Protocol [44] messages over A2A or ACP). During negotiation, agents exchange more detailed VCs about their specific sub-capabilities, current availability, and resource needs.
- The orchestrator then issues a "CoalitionCharter\_VC" to the selected agents, defining the coalition's DID, its mission, shared resources (perhaps managed by a temporary group DID), roles, and duration. This VC acts as a temporary authorization within the coalition.

#### *ANS for Discovering Ethical AI Governance Services:*

- Agents or users could query ANS for services like [audit://EthicalComplianceOracle.AIBehavior.IndependentOrg.v1](#) or [report://BiasReportingService.FairnessConsortium.v1](#).
- These specialized services (themselves having DIDs and VCs) could then be used by agents to self-assess their decisions against ethical guidelines or for users to report problematic agent behavior, with the AN DIDs providing a verifiable link to the service.

By integrating ANS as a core discovery mechanism whose results (DIDs, initial capability claims) feed directly into the DID/VC-based authentication and authorization processes, the entire IAM lifecycle becomes more context-aware, secure, and efficient. The discoverable nature of agent capabilities and attestations fosters a more transparent and trustworthy ecosystem.

## VI. DEPLOYMENT MODELS & GOVERNANCE CONSIDERATIONS

The proposed Agentic AI IAM framework, while architecturally comprehensive, is not a monolithic, one-size-fits-all solution in terms of its practical implementation. The diverse needs of different organizations, Multi-Agent System (MAS) scopes (private enterprise vs. open ecosystem), trust requirements, and existing infrastructure will necessitate different deployment models for its core components (e.g., DID registries, Verifiable Credential (VC) issuers, Agent Name Service (ANS), Policy Engines, Session Authority, Session State Synchronizer). Furthermore, regardless of the chosen deployment model, robust, well-defined, and adaptable governance is paramount for the long-term viability, trustworthiness, security, and interoperability of any such advanced IAM system.

### *A. Deployment Model Analysis*

We analyze three primary deployment models—Centralized, Decentralized, and Federated—assessing their characteristics, advantages, disadvantages, and suitability for various Agentic AI IAM scenarios.

*1) Centralized Approach: Description:* In a centralized deployment, a single organization, platform provider, or a designated administrative entity controls and operates all, or the significant majority, of the IAM framework's core components. This typically includes: The primary Agent ID registry (which might be a private Public Key Infrastructure (PKI) issuing X.509 certificates as per some ANS proposals, a proprietary database issuing unique identifiers, or a private DID method controlled by the organization). The authoritative VC issuers for organizational roles, capabilities, and compliance attestations. The ANS, if implemented as a private or enterprise-scoped directory service. The central Policy Decision Points (PDPs) and Policy Administration Points (PAPs) defining and enforcing access rules. The Cross-Protocol Session Authority (SA) and the Session State Synchronizer (SSS). Agents operating within this model typically belong to, or are tightly managed and permissioned by, the central entity. All trust decisions ultimately flow from this central authority.

*Advantages:* Simplified Governance & Policy Cohesion. Unified Control, Visibility, and Audit. Potentially Easier Integration with Existing Enterprise Systems. Optimized Performance. Clear Accountability.

*Disadvantages:* Single Point of Failure, Control, and Trust. Scalability Bottlenecks. Vendor/Platform Lock-in. Limited Cross-Organizational Trust & Interoperability. Potential for Censorship or Abuse of Power.

*When to Use:* Enterprise-Internal MAS. Specific AI Platforms. Early-Stage Deployments or Controlled Experiments. Highly Regulated Environments with a Single Auditing Authority.

*2) Decentralized Approach: Description:* Core IAM components are implemented using decentralized technologies, often public and permissionless, or permissioned consortia-based Distributed Ledger Technologies (DLTs). Key characteristics include: DIDs are registered on public or consortia DLTs (e.g., did:ion, did:ethr, did:sov, or a custom agent-focused DID method on a dedicated ledger like the proposed Agent ID Provider Network - AIPN). Agent controllers or agents themselves manage their DID's private keys. VCs can be issued by a diverse set of issuers (each with their own DID) and their status (revocation) might be tracked via decentralized mechanisms (e.g., on-chain registries, distributed VC status lists). ZKPs are used extensively for privacy-preserving presentation of VCs and attributes. ANS could be built on decentralized name systems (e.g., ENS, Handshake, or a custom DLT-based ANS). Policy enforcement might involve smart contracts acting as rudimentary PDPs for on-chain resources, or rely on Verifiable Presentations that bundle VCs required by a verifier's policy. Global session state (like revocation lists) might be mirrored on resilient DLTs. Governance is typically community-driven (e.g., DAOs for protocol upgrades) or based on the immutable logic encoded in smart contracts.

*Advantages:* No Single Point of Failure or Control. User/Agent Sovereignty (SSI). Enhanced Trust in Open, Permissionless Ecosystems. Transparency & Auditability (for public DLTs). Censorship Resistance.

*Disadvantages:* Governance Complexity, and "Tragedy of the Commons". Smart Contract and DLT Security Risks. Performance, Scalability, and Cost of DLTs. User/Controller Experience (Key Management). Irreversibility and Data Privacy. Bootstrapping Trust.

*When to Use:* Truly Open, Permissionless Multi-Agent Ecosystems. Cross-Organizational Collaborations Without a Central Trusted Party. Applications Requiring Very High Degrees of Censorship Resistance or User Control Over Identity. Ecosystems Where a Transparent, Community-Governed Trust Infrastructure is a Core Design Goal.

*3) Federated Approach: Description:* This model involves multiple independent IAM domains or "trust communities." Each domain might manage its own IAM infrastructure using centralized or even localized decentralized approaches. The key is that these domains establish mutual trust relationships and define standardized protocols for interoperability. This could involve: Cross-certification of Certificate Authorities (CAs) or DID method roots between domains. Shared trust lists for recognized VC issuers and verifier policies across the federation. Federated ANS resolution (e.g., similar to how DNS subdomains can be delegated, or using inter-registry lookup protocols). Use of highly interoperable DID methods and standardized VC profiles (e.g., based on W3C specs) to ensure credentials from one domain can be understood and verified in another. A central (or mutually agreed upon) body might define the "federation rules" or baseline interoperability standards, but day-to-day IAM within each domain remains autonomous.

*Advantages:* Balances Autonomy with Interoperability. Scalability. Domain-Specific Policies and Trust Levels. Enhanced Resilience. Phased Adoption & Existing System Integration.

*Disadvantages:* Complexity of Trust Management. Interoperability Challenges (Technical and Semantic). Potential for Lowest Common Denominator Security. Discovery and Pathfinding Complexity. Governance Overhead for the Federation Itself.

*When to Use:* Consortia of Organizations in a Specific Industry. Alliances of Research Institutions or Governmental Agencies. Large, Multi-National Corporations with Distinct Regional or Business Unit IAM Requirements. Ecosystems Evolving from Existing Centralized or Siloed Systems Towards Greater Interoperability. As a practical model for the Agent Name Service (ANS).

*4) Hybrid Approaches:* It's important to note that these models are not always mutually exclusive. Hybrid approaches are likely to be common, combining elements from each.

- *Example 1:* An enterprise might use a centralized IAM framework for its internal agents but use a federated model to interact with agents from trusted partners. Its internal agents might have DIDs issued by a private DID method, but these DIDs could be anchored or discoverable through a broader federated system.
- *Example 2:* A decentralized ecosystem might still rely on a few, highly reputable (perhaps foundation-run) "anchor" VC issuers for certain critical credentials (like

”VerifiedLegalEntity\_VC”), even if most other VCs are issued more peer-to-peer.

- *Example 3:* The Session Authority and Session State Synchronizer, while logically providing global coordination, might be implemented as a permissioned DLT operated by a consortium (federated control over a logically centralized function) for resilience and shared trust.

### B. Decision Matrix for Choosing an Implementation Model

Selecting the most appropriate deployment model requires careful consideration of various factors. Table II provides guidance:

*How to use the matrix:*

- 1) Identify the primary context for your MAS (e.g., internal enterprise, open research platform, industry consortium).
- 2) Prioritize your key requirements (e.g., is maximum agent sovereignty critical, or is centralized auditability paramount?).
- 3) Evaluate each model against your high-priority requirements.
- 4) Consider if a hybrid approach offers the best trade-offs by combining strengths of different models for different IAM components (e.g., decentralized DIDs but a federated or even centrally managed Session Authority for specific use cases).

### C. Governance Considerations

Effective governance is the bedrock upon which trust and interoperability in any Agentic AI IAM framework are built. It’s not merely about technical rules but also about establishing clear roles, responsibilities, processes for decision-making, dispute resolution, and adaptation over time.

#### 1) Identity Governance (DIDs, VCs, ANS):

- DID Method Governance
- ANS Namespace Management & Policy
- VC Issuer Accreditation, Trust Registries, and Governance Frameworks
- Agent ID Lifecycle Management Policies

#### 2) Security Policy Governance (for PDPs and SA):

- Policy Authorship & Approval Workflows
- Policy-as-Code Principles
- Emergency Policy Override Procedures
- Policy Interoperability/Harmonization (in Federated Models)

#### 3) Operational and Security Governance for IAM Infrastructure:

- Incident Response Playbooks for IAM Breaches
- Key Management Governance for IAM Services
- Regular Audits & Penetration Testing
- Vulnerability Disclosure Policy

#### 4) Data Privacy and Ethical Use Governance:

- Data Protection Impact Assessments (DPIAs) for IAM Data
- Agent ID Data Minimization Principles
- Bias Review in Credentialing and Reputation
- Ethical Oversight Bodies

#### 5) Evolution and Standards Governance:

- Change Management Process
- Liaison with External Standards Bodies

Effective governance in the Agentic AI IAM space will not be static; it must be an adaptive system capable of evolving alongside the technology and the threat landscape. It necessitates a collaborative effort, potentially involving a mix of industry self-regulation, standards development, and, where appropriate, governmental oversight, particularly for public-facing or critical infrastructure components.

## VII. SECURITY CONSIDERATIONS

Securing the Agentic AI IAM framework is paramount, analyzed here using the MAESTRO framework [45].

### A. The MAESTRO 7-Layer Reference Architecture for Agentic AI

MAESTRO decomposes AI ecosystems into: Layer 1: Foundation Models, Layer 2: Data Operations, Layer 3: Agent Frameworks, Layer 4: Deployment and Infrastructure, Layer 5: Evaluation and Observability, Layer 6: Security and Compliance (Vertical), and Layer 7: Agent Ecosystem.

### B. Threat Analysis of the Proposed Agentic AI IAM Framework using MAESTRO Layers

- **L1: Foundation Models:** Model-based identity theft that occurs when attackers use AI models to analyze and replicate the behavioral patterns, communication styles, and decision-making characteristics of legitimate agents, effectively creating digital impersonators that can fool other systems or users into believing they’re interacting with the authentic agent (mitigated by cryptographic DIDs/VCs).
- **L2: Data Operations:** Poisoning of DID registries/VC status lists (mitigated by DLT consensus, signed registry entries); exfiltration of identity data (mitigated by encryption, agent-held VCs, ZKPs); tampering with PIPs (mitigated by PIP identity and secure channels).
- **L3: Agent Frameworks:** Compromised IAM SDKs (mitigated by secure development, sandboxing); framework vulnerabilities allowing session hijacking (mitigated by continuous re-validation via AEM/SSS).
- **L4: Deployment and Infrastructure:** DoS/DDoS against IAM services (mitigated by standard defenses, resilient design); compromise of IAM service infrastructure (mitigated by hardening, access controls); lateral movement to IAM components (mitigated by network segmentation, Zero Trust).
- **L5: Evaluation and Observability:** Tampering with IAM audit logs (mitigated by immutable logging, signatures); evasion of IAM monitoring (mitigated by comprehensive instrumentation); data leakage via observability tools (mitigated by masking, ZKPs).
- **L6: Security and Compliance:** Misconfiguration of IAM policies (mitigated by policy-as-code, audits); compromise of IAM service keys (mitigated by HSMs, revocation);



TABLE II  
DECISION MATRIX FOR CHOOSING AN IMPLEMENTATION MODEL

Feature / Requirement	Centralized	Decentralized	Federated	Hybrid
Control Authority	Single Entity (High Control)	Community/Protocol (Low Central Control)	Domain-Specific + Federation Body (Balanced)	Varies; often domain-specific with shared elements
Trust Model	Hierarchical (Trust in Central Entity)	Peer-to-Peer / Ledger-Based (Distributed Trust)	Inter-Domain Agreements / Shared Roots (Delegated)	Mix of hierarchical and delegated/distributed
Scalability	Moderate (Potential Bottlenecks)	Potentially Very High (if DLT scales) / Variable	High (Distributed across domains)	High (Can optimize components)
Performance (Latency)	Potentially Low (if optimized, local)	Variable (DLT dependent, often higher)	Moderate (Inter-domain hops)	Variable (Can optimize critical paths)
Interoperability (External)	Low (Proprietary by default)	Potentially High (if open standards used)	High (Designed for inter-domain ops)	Moderate to High (Depends on bridge design)
Complexity of Setup	Low to Moderate	High	High (Trust agreements complex)	Moderate to High
Complexity of Governance	Low (Single decision-maker)	Very High (Consensus, community)	High (Federation rules, inter-domain)	High (Managing diverse components)
Cost (Infrastructure)	Moderate (Centralized infra)	Variable (DLT fees can be high)	Moderate to High (Per-domain + federation infra)	Variable
Security (vs External Threats)	Single attack surface (high impact if breached)	Distributed risk, smart contract vulns critical	Risk shared/isolated per domain; inter-domain trust	Tailorable; can have strong internal, defined external
User/Agent Sovereignty	Low	Very High	Moderate (Within domain policies)	Variable
Censorship Resistance	Low	High	Moderate (Per domain)	Variable
Privacy Preservation	Dependent on central entity's policies	High (with ZKPs, careful DLT use)	Domain-specific policies; inter-domain data flow	Can be designed for high privacy
Suitability: Enterprise Internal	High	Low	Moderate (For large, distinct internal units)	High (Central core, federated edges)
Suitability: Open Ecosystem	Low	High	Moderate (Federation of open communities)	Moderate (Public services with private backends)
Suitability: B2B Consortia	Low (Unless one org dominates)	Moderate (If common DLT agreed)	High	High (Federated interfaces, shared services)

non-compliance with privacy regulations (mitigated by privacy-by-design, ZKPs).

- **L7: Agent Ecosystem:** Agent impersonation/DID spoofing (mitigated by cryptographic verification, VC status checks); compromised ANS leading to malicious discovery (mitigated by secure ANS resolution); collusion to falsify VCs (mitigated by trust diversification, reputation systems).

#### C. Cross-Layer Threats Affecting the IAM Framework

Including supply chain attacks on IAM components, privilege escalation across IAM layers, and goal misalignment leading to IAM misuse, all requiring defense-in-depth and continuous monitoring.

#### D. Applying Zero Trust to Agentic AI IAM Framework

This brings essential security, governance, and accountability benefits especially given the autonomous decision making, undeterministic behavior, and scale of AI agents. Implemented and tested security controls that are preventative, detective, and corrective form the basis of Zero Trust. These fundamentals are critical to the success of a Zero Trust implementation: Concept of least-privilege access, Separation of duties, Segmentation/micro-segmentation, Logging and monitoring, Configuration drift remediation, Assume breach, Dynamic and adaptive security policy enforcement.

#### VIII. INNOVATIVE CONTRIBUTIONS OF THIS FRAMEWORK

The proposed framework represents a significant departure from traditional approaches, offering a collection of synergistic innovations specifically designed for the unique challenges of autonomous Multi-Agent Systems (MAS). These contributions are not isolated features but form part of a re-conceptualization of agent identity, integrating advanced cryptographic techniques and a novel architectural design for dynamic control, all within a holistic, lifecycle-aware approach to managing AI agents as first-class digital citizens.

The foremost contribution is the articulation of a comprehensive, end-to-end IAM framework purpose-built for the agentic paradigm. This moves beyond merely adapting human-centric or simplistic machine and NHI (Non Human Identity) IAM protocols, which often prove inadequate for the complexities of autonomous, interacting agent swarms. Instead, our framework cohesively integrates identity issuance, rich credentialing, capability-aware discovery, dynamic access control, and a novel cross-protocol enforcement layer into a unified conceptual model. It addresses the entire lifecycle of an agent—from its "birth" through its operational interactions to its eventual decommissioning—recognizing the deep interdependencies between these stages. Existing IAM solutions typically focus on narrower problems, struggling with identities that spawn others, dynamically change roles, or require fine-grained, context-sensitive authorization at massive scale. This framework's

systemic integration is designed to address these fundamental gaps.

Central to this is a redefinition of Agent Identity, making it rich, dynamic, and verifiably secure. We shift away from simplistic identifiers like API keys towards identities anchored by cryptographically secure Decentralized Identifiers (DIDs). This DID-anchored identity is not static; it is an extensible digital representation augmented by Verifiable Credentials (VCs) that attest to an agent’s attributes, capabilities, compliance status, roles, and provenance. The dynamism is crucial, as AI agents evolve, their models update, capabilities expand, and compliance needs re-attestation. A rich, verifiable identity containing fields like `scopeOfBehavior`, `toolset` (which can include DIDs of authorized tools), `modelHash`, and VCs for training data or compliance, allows for far more nuanced trust and authorization. The use of DIDs provides self-sovereignty and interoperability, essential for open MAS, while VCs offer a standardized, vendor-neutral way to make diverse claims. Furthermore, Zero-Knowledge Proofs (ZKPs) enable agents to selectively and privately present these verifiable claims, a significant advancement over the limited flexibility and privacy of traditional certificate extensions.

Building on this rich identity, the framework introduces capability-centric discovery and more granular access control. An integrated Agent Naming Service (ANS) facilitates secure discovery, allowing agents to find others not just by name but by the specific functions or attested capabilities they offer. This is a critical distinction from traditional service discovery, which may locate an endpoint but doesn’t inherently verify the target’s attested abilities. Our approach directly links discovery to verifiable identity attributes. Authorization decisions thereby become more intelligent, considering not just “who” is making a request, but fundamentally “what is this agent verifiably capable and authorized to do, with which specific tools, and under what attested conditions?” By making an agent’s authorized toolset and `scopeOfBehavior` verifiable parts of its identity, the system can enforce the principle of least function, significantly limiting the blast radius of a compromised or misbehaving agent. The framework introduces Context-Based Access Control which enables dynamic access decisions based on real-time environmental, behavioral, and task context moving beyond static roles or attributes allowing enforcement policies to adapt to an agent’s current state and conditions.

A cornerstone innovation is the Unified Cross-Protocol Global Session Management and Policy Enforcement Architecture. This Layer 4 uniquely addresses the challenge of maintaining consistent security posture in heterogeneous MAS where agents use diverse communication protocols. In such environments, a critical security gap is the inability to propagate vital IAM state changes—like a global session termination, a master DID revocation, or a sudden capability downgrade—instantaneously and uniformly across all interaction points. This layer acts as a “security and session management backplane,” ensuring that a policy decision or revocation, once made, is effectively and immediately enforced wherever an agent might interact, regardless of the underlying transport.

This real-time, cross-protocol consistency is fundamental for operationalizing robust security.

The framework also achieves a pragmatic fusion of self-sovereignty with enforceable governance. While DIDs and agent-controlled VCs empower agents and their controllers with greater control over their core identity data, this self-sovereignty is balanced with mechanisms for practical governance. This means that while an agent can present its self-managed identity, these credentials can be verified against established trust frameworks, such as lists of accredited VC issuers for specific roles or compliance attestations. The Session Authority retains the ability to enforce global revocations or policy overrides based on enterprise risk decisions, even if the agent “controls” its DID. This balance is vital for adoption in real-world systems that require clear lines of accountability and cannot operate solely on peer-to-peer trust.

Finally, the framework provides intrinsic support for fine-grained accountability and verifiable provenance. Cryptographic verifiability is embedded at multiple levels: for identities via DIDs and their keys; for claims about agents via VCs and issuer signatures; for agent actions using the agent’s DID-associated private key. The Agent ID structure itself is designed to encapsulate or link to detailed provenance information—such as its creator, constituent models, software dependencies (potentially with their own DIDs), and VCs attesting to training data or safety audits. As AI agents are entrusted with increasingly impactful decisions, the ability to irrefutably determine “who (which agent instance) did what, when, why, with what authority, and based on what information/capabilities” becomes critical. This moves beyond basic logging to establish a cryptographically verifiable audit trail, essential for forensics, dispute resolution, and building societal trust in autonomous systems, directly addressing the “audit trail ambiguity” prevalent in current systems and providing a much stronger basis for non-repudiation.

To measure the success implementation of the innovation, the following Key Performance Indicators (KPIs) can be considered: Successful Agent Authentication Rate, Authorization Latency, Policy Enforcement Accuracy, Revocation Time, Audit Log Integrity, Anomaly Detection Rate, Incident Response Time, Agent Discovery Success Rate, Downtime due to IAM Issues.

## IX. DISCUSSION AND FUTURE WORK

As future work, we have identified the following.

### A. Scalability, Performance, and Efficiency

The Challenge: Several components within the proposed architecture, particularly those involving Distributed Ledger Technologies (DLTs) for DID registration and Verifiable Credential (VC) status management, or the Session State Synchronizer (SSS) which must track potentially millions of active agent sessions, face significant scalability and performance hurdles. The cryptographic operations inherent in DIDs, VCs, and Zero-Knowledge Proofs (ZKPs), while providing security, can also introduce computational overhead for resource-constrained agents or high-throughput systems.

Future Work: Benchmarking and Optimization; Efficient Cryptography; Caching and Resolution Strategies; Hardware Acceleration.

### *B. Standardization and Interoperability*

The Challenge: The true power of a global Agentic AI IAM framework lies in its interoperability. Without widely adopted standards for how Agent IDs are structured, how capabilities are defined and attested in VCs, how ZKPs are constructed for common proofs, or how ANS queries are formatted, the ecosystem risks fragmentation into incompatible identity silos.

Future Work: Active Standards Development; Agent-Specific Profiles; Common Ontologies; Reference Implementations and Conformance Suites; Formalization of Model Context Protocols (MCPs).

### *C. Governance Models, Trust Frameworks, and Legal Considerations*

The Challenge: Establishing and managing governance for a potentially global, decentralized, or federated IAM infrastructure is a monumental task. This includes defining who can issue authoritative VCs (e.g., for legal identity or compliance), how disputes over DIDs or ANS names are resolved, and how liability is attributed in complex MAS interactions. The evolving legal and regulatory landscape for AI also presents a moving target.

Future Work: Multi-Stakeholder Governance Research; Trust Assurance Levels; Legal and Regulatory Analysis; Dispute Resolution Mechanisms; Security Controls Specific to AI Agents.

### *D. Enhanced Security and Privacy in Practice*

The Challenge: While the framework incorporates strong security primitives, sophisticated adversaries will inevitably seek to exploit implementation weaknesses, social engineering aspects, or unforeseen interaction effects between components. Maintaining agent and user privacy in the face of increasingly rich identity data is also paramount.

Future Work: Formal Security Modeling and Verification; Agent-Specific Threat Intelligence; Advanced Privacy-Enhancing Technologies (PETs); Secure Key Management for Autonomous Agents; Resilience Against Quantum Threats; Tabletop Exercises for Agentic Incident Response.

### *E. User Experience (UX), Developer Tooling, and Adoption Pathways*

The Challenge: For this framework to be adopted, it must be usable by both end-users (who may act as controllers for their personal agents) and developers building and deploying AI agents. Complexity in managing DIDs, VCs, and policies can be a significant barrier.

Future Work: Developer-Friendly SDKs and Libraries; Management UIs and Dashboards; "Secure by Default" Agent Architectures; Phased Adoption Strategies.

### *F. Ethical Considerations and Societal Impact Mitigation*

The Challenge: The power of verifiable and persistent Agent IDs, while beneficial for security, also carries potential risks if misused for pervasive surveillance, biased decision-making (e.g., if VCs for "good behavior" are only available to certain types of agents), or creating new forms of digital divide.

Future Work: Ethical Impact Assessments; Bias Detection and Mitigation in Credentialing; Transparency and Explainability of IAM Decisions; Public Discourse and Inclusive Design.

The journey to a fully realized and globally functional Agentic AI IAM framework is an ambitious one. It necessitates a collaborative, iterative approach, blending cutting-edge research with pragmatic engineering and a deep understanding of the evolving societal context of AI. Addressing these future work areas will be critical to transforming the vision presented in this paper into a resilient, trustworthy, and enabling infrastructure for the future of AI.

### ACKNOWLEDGMENT

The authors extend their deepest gratitude to the following individuals whose expertise, insights, and collaborative spirit made this research on Agentic AI Identity and Access Management approaches possible.

*Authors and Contributors (implicitly covered by the author list, but acknowledged as per original text):* Vineeth Sai Narajala, John Yeoh, Json Ross, Mahesh Lambe, Ramesh Raskar, Youssef Harkati, Jerry Huang, Chris Hughes.

*Reviewer:* Idan Habler, PhD, Staff AI/ML Security Researcher at Intuit, whose thorough review and constructive feedback significantly improved the quality and accuracy of this research.

*Brainstorming Contributors:* Special appreciation goes to the following thought leaders who have discussed with Ken Huang on Agentic AI security on many different occasions in his involvement as co-chair of the CSA AI Safety Working groups, at the RSA 2025 conferences, and LinkedIn discussions: Jim Reavis, CEO and Founder of Cloud Security Alliance; Daniele Catteddu, Chief Technology Officer at Cloud Security Alliance; Caleb Sima, Chair of CSA AI Security Alliance; Professor Dawn Song of the University of California, Berkeley; Michael Bargury, Co-founder and CTO of Zenity; Dr. Chenxi Wang of Rain Capital; Nate Lee, Founder of Cloudsec.ai; Ed Sewell, NVIDIA AI INCEPTION Member; Jojo Ye of Sixty Degree Capital; Akram Sheriff of Cisco.

This research on Agentic AI Identity and Access Management stands on the shoulders of these remarkable individuals, whose collective wisdom, diverse perspectives, and unwavering support made this work possible. Their contributions reflect the collaborative spirit essential to advancing Agentic AI Security.

### REFERENCES

- [1] K. Huang, "Agentic AI identity management approach," Cloud Security Alliance Blog, March 2025. [Online]. Available: <https://cloudsecurityalliance.org/blog/2025/03/11/agentic-ai-identity-management-approach>

- [2] V. C. Hu, D. F. Ferraiuolo, D. R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller, and K. Scarfone, "Guide to attribute based access control (abac) definition and considerations," National Institute of Standards and Technology, Tech. Rep. NIST SP 800-162, 2019. [Online]. Available: <https://doi.org/10.6028/NIST.SP.800-162>
- [3] N. Yaqub, J. Zhang, M. I. Khalid, W. Wang, M. Helfert, M. Ahmed, and J. Kim, "Blockchain enabled policy-based access control mechanism to restrict unauthorized access to electronic health records," *PeerJ Computer Science*, vol. 11, p. e2647, 2025. [Online]. Available: <https://doi.org/10.7717/peerj-cs.2647>
- [4] V. Shastri, "What is just-in-time (jit) access?" CrowdStrike, January 2025, accessed: May 23, 2025. [Online]. Available: <https://www.crowdstrike.com/en-us/cybersecurity-101/identity-protection/just-in-time-access/>
- [5] J. Ferber, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, 1999.
- [6] European Parliament and Council, "Regulation of the European Parliament and of the Council on Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts," Official Journal of the European Union, 2023.
- [7] S. Rosenbush, "AI Agents Face One Last, Big Obstacle," *The Wall Street Journal*, May 2025.
- [8] A. Chan *et al.*, "IDs for AI Systems," *arXiv preprint arXiv:2406.12137*, 2024.
- [9] H. Ken, V. S. Narajala, I. Habler, and A. Sheriff, "Agent name service (ans): A universal directory for secure ai agent discovery and interoperability," *arXiv preprint arXiv:2505.10609*, 2025. [Online]. Available: <https://arxiv.org/abs/2505.10609>
- [10] D. E. Hardt, "The OAuth 2.0 Authorization Framework," Internet Engineering Task Force, RFC RFC 6749, 2012. [Online]. Available: <https://doi.org/10.17487/RFC6749>
- [11] OpenID Foundation, "OpenID Connect Core 1.0 incorporating errata set 1," OpenID Foundation Standards, 2014. [Online]. Available: [https://openid.net/specs/openid-connect-core-1\\_0.html](https://openid.net/specs/openid-connect-core-1_0.html)
- [12] OASIS, "Security Assertion Markup Language (SAML) V2.0 Errata," OASIS Standard, 2005.
- [13] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, "The kerberos network authentication service (v5)," Internet Engineering Task Force, RFC RFC 4120, July 2005, accessed: May 23, 2025. [Online]. Available: <https://datatracker.ietf.org/doc/html/rfc4120>
- [14] J. Sermersheim, "Lightweight directory access protocol (ldap): The protocol," Internet Engineering Task Force, RFC RFC 4511, June 2006, accessed: May 23, 2025. [Online]. Available: <https://datatracker.ietf.org/doc/html/rfc4511>
- [15] Anthropic, "Introducing the model context protocol," Anthropic News, November 2024, accessed: May 23, 2025. [Online]. Available: <https://www.anthropic.com/news/model-context-protocol>
- [16] modelcontextprotocol (Organization), "Specification and documentation for the model context protocol," GitHub, 2025, accessed: May 23, 2025. [Online]. Available: <https://github.com/modelcontextprotocol/modelcontextprotocol>
- [17] V. S. Narajala and I. Habler, "Enterprise-Grade Security for the Model Context Protocol (MCP): Frameworks and Mitigation Strategies," *arXiv preprint arXiv:2504.08623*, 2025. [Online]. Available: <https://arxiv.org/abs/2504.08623>
- [18] H. Koshutanski and K. Hristov, "Enhancing grid security by fine-grained behavioral control and negotiation-based authorization," *International Journal of Information Security*, vol. 7, no. 5, pp. 327–341, 2008.
- [19] M. Thompson, A. Essiari, and S. Mudumbai, "Solving the transitive access problem for the services oriented architecture," in *Proceedings of the IEEE International Conference on Web Services (ICWS 2007)*, 2007, pp. 390–397.
- [20] A. Bouhairie and A. Hair, "SCPAC: An Access Control Framework for Diverse IoT Platforms Based on OAuth2.0," in *2021 International Conference on Promising Electronic Technologies (ICPET)*, 2021, pp. 1–6.
- [21] S. Ren, S. M. Roy, Y. Gao, K. Nahrstedt, and R. Wang, "Next generation session management for 3d teleimmersive interactive environments," *Multimedia Tools and Applications*, vol. 54, no. 3, pp. 545–577, 2009.
- [22] R. A. Shaikh, E. Adi, and H. Koshutanski, "Trust and identity management in cloud and distributed systems," in *Security in Computing and Communications (SSCC 2015)*, ser. Communications in Computer and Information Science, X. Lin and Y. Xiang, Eds. Springer, 2015, vol. 536, pp. 3–16.
- [23] L. Choda, "A practitioner's guide to managing non-human identity risks," Video, KuppingerCole Analysts, May 2025, accessed: May 23, 2025. [Online]. Available: <https://www.kuppingercole.com/watch/practitioner-s-guide-non-human-risks-eic25>
- [24] D. Fett, R. Kuesters, and G. Schmitz, "A comprehensive formal security analysis of oauth 2.0," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*, 2016, pp. 1204–1215.
- [25] SSH Communications Security, "A guide to keyless and passwordless authentication," SSH Communications Security Website, n.d., retrieved May 23, 2025. [Online]. Available: <https://www.ssh.com/academy/secret-s-management/keyless-passwordless-authentication-guide>
- [26] World Wide Web Consortium (W3C), "Decentralized Identifiers (DIDs) v1.0," World Wide Web Consortium (W3C), W3C Recommendation, July 2022. [Online]. Available: <https://www.w3.org/TR/did-core/>
- [27] —, "Verifiable Credentials Data Model v1.0," World Wide Web Consortium (W3C), W3C Recommendation, November 2021. [Online]. Available: <https://www.w3.org/TR/vc-data-model/>
- [28] M. Sporny *et al.*, "Verifiable Credentials Data Model v2.0," World Wide Web Consortium (W3C), W3C Working Draft, May 2024. [Online]. Available: <https://www.w3.org/TR/vc-data-model-2.0/>
- [29] K. Huang, "Agentic AI DID SDK: Decentralized Identifiers and Zero-Knowledge Proofs," GitHub Repository, 2025. [Online]. Available: <https://github.com/kenhuangus/agent-id-sdk>
- [30] "OWASP Top 10 Non-Human Identities Risks - 2025," OWASP Foundation Report, OWASP, 2025. [Online]. Available: <https://owasp.org/www-project-non-human-identities-top-10/2025/top-10-2025/>
- [31] "The State of Non-Human Identity Security," Cloud Security Alliance Survey Report, Cloud Security Alliance, 2024. [Online]. Available: <https://cloudsecurityalliance.org/artifacts/state-of-non-human-identity-security-survey-report>
- [32] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186–208, 1989.
- [33] K. Huang, V. S. Narajala, I. Habler, and A. Sheriff, "Agent Name Service (ANS): A universal directory for secure AI agent discovery and interoperability," Internet Engineering Task Force, Internet-Draft draft-narajala-ans-00, May 2025. [Online]. Available: <https://datatracker.ietf.org/doc/draft-narajala-ans/>
- [34] OASIS, "extensible access control markup language (xacml) version 3.0," OASIS Standard, Tech. Rep., January 2013, accessed: May 23, 2025. [Online]. Available: <http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html>
- [35] Open Policy Agent, "Policy language," Open Policy Agent Documentation, 2025, accessed: May 23, 2025. [Online]. Available: <https://www.openpolicyagent.org/docs/policy-language/>
- [36] J. Kindervag, "Build Security Into Your Network's DNA: The Zero Trust Network Architecture," Forrester Research, Tech. Rep., 2010.
- [37] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," National Institute of Standards and Technology, Tech. Rep. NIST SP 800-207, August 2020. [Online]. Available: <https://doi.org/10.6028/NIST.SP.800-207>
- [38] E. G. Junior, S. Clinton, C. Hughes, V. S. Narajala, and T. Holmes, "LLM and GenAI data security best practices," Feb. 2025. [Online]. Available: [https://www.researchgate.net/publication/391204648\\_LLM\\_and\\_GenAI\\_Data\\_Security\\_Best\\_Practices](https://www.researchgate.net/publication/391204648_LLM_and_GenAI_Data_Security_Best_Practices)
- [39] laxmih (Google Cloud Community), "Understanding a2a — the protocol for agent collaboration," Google Cloud Community Blog, May 2025, accessed: May 23, 2025. [Online]. Available: <https://www.googlecloudcommunity.com/gc/Community-Blogs/Understanding-A2A-The-Protocol-for-Agent-Collaboration/ba-p/906323>
- [40] A. Ehtesham, A. Singh, G. K. Gupta, and S. Kumar, "A survey of agent interoperability protocols: Model context protocol (mcp), agent communication protocol (acp), agent-to-agent protocol (a2a), and agent network protocol (anp)," *arXiv preprint arXiv:2505.02279*, 2025.
- [41] K. Huang, A. Sheriff, J. Sotiropoulos, R. F. Del, and V. Lu, "Multi-agentic system threat modelling guide OWASP GenAI security project," Apr. 2025. [Online]. Available: [https://www.researchgate.net/publication/391204915\\_Multi-Agentic\\_System\\_Threat\\_Modelling\\_Guide\\_OWASP\\_GenAI\\_Security\\_Project](https://www.researchgate.net/publication/391204915_Multi-Agentic_System_Threat_Modelling_Guide_OWASP_GenAI_Security_Project)
- [42] V. S. Narajala, K. Huang, and I. Habler, "Securing genai multi-agent systems against tool squatting: A zero trust registry-based approach," *arXiv preprint arXiv:2504.19951*, 2025. [Online]. Available: <https://arxiv.org/abs/2504.19951>



- [43] I. Habler, K. Huang, V. S. Narajala, and P. Kulkarni, "Building a secure agentic AI application leveraging A2A protocol," 2025. [Online]. Available: <https://www.arxiv.org/abs/2504.16902>
- [44] FIPA TC C, "Fipa contract net interaction protocol specification," Foundation for Intelligent Physical Agents, Tech. Rep. FIPA Standard SC00029H, December 2002, accessed: May 23, 2025. [Online]. Available: <http://fipa.org/specs/fipa00029/SC00029H.html>
- [45] K. Huang, "Agentic AI Threat Modeling Framework: MAESTRO," Cloud Security Alliance Blog, February 2025, uRL not provided in source. Accessed: May 23, 2025.