

Response to NIST Request for Information Regarding Security Considerations for Artificial Intelligence Agents (NIST-2025-0035)

Submission contributors: OpenID Foundation's [AIIM Community Group](#), Threat Modelling sub-group

Date: March 6, 2026

I. Introduction & Organizational Context

The OpenID Foundation (OIDF) is a non-profit international standardization organization dedicated to promoting, protecting, and nurturing the OpenID community and technologies. For nearly two decades, the Foundation has developed and maintained identity standards that underpin authentication and authorization across the internet.

This response is submitted on behalf of the OIDF AI Identity Management (AIIM) Community Group, specifically the Threat Modeling Subgroup. The AIIM Community Group was established to address the emerging intersection of artificial intelligence systems and digital identity infrastructure. Our membership includes practitioners from major technology companies, financial institutions, security vendors, and independent researchers who work daily with identity systems at scale.

The Threat Modeling Subgroup focuses on identifying and cataloging security risks that arise when AI agent systems interact expecting detailed, fine-grained results from the identity and access management infrastructure layer which is still maturing in its use and deployment. This mismatch between expectations by both people(users) and the technology(agents, the AI itself) and ground truth in what has been implemented is the gap and our focus area. The AIIM Community Group recognized the challenges of identity and [AI early publishing a white paper October 7, 2025](#) as a call to action to delve deeper into the topics and for which some of our answers for the NIST RFI build upon. Our work draws from a community of identity and standards practitioners and uses established frameworks including MITRE ATT&CK, and the OWASP Top 10 for Generative AI Applications, adapted for the specific challenges that agentic AI presents.

We believe identity infrastructure represents a critical and underexamined dimension of AI agent security. The questions NIST poses in this RFI regarding threats, security practices, assessment methods, and deployment environments cannot be adequately answered without addressing how agents authenticate, how they obtain and use credentials, and how their actions can be attributed to responsible parties. Current practices in the AI ecosystem frequently violate foundational security principles that the identity community established decades ago, creating urgent risks that will compound as agent deployment accelerates.

Many of these risks are not failures of token formats or encryption algorithms. They are failures of trust governance: ad hoc allowlists, unsigned metadata, manual key registration, and weak revocation paths. As agent ecosystems become multi-party and cross-domain, these approaches do not scale. This response therefore distinguishes between (1) authorization primitives (OAuth/OIDC), (2) enforcement architecture (PDP/PEP and gateways), and (3) the trust fabric that determines which issuers, tools, and metadata are trustworthy in the first place.

The observations and recommendations in this response emerge from ongoing threat modeling work within our subgroup, informed by real-world implementations our members have observed across regulated industries including financial services, healthcare, and government contracting.

II. Response to Section 1: Security Threats, Risks, and Vulnerabilities

2.1 Unique Security Threats Distinct from Traditional Software Systems (Question 1a)

The most pressing security threat affecting AI agent systems today is not a novel technical exploit but a widespread practice: credential sharing between humans and agents. Across industries, we observe users granting AI agents direct access to corporate systems by sharing their own credentials, passwords, API keys, OAuth tokens, and session cookies. This practice is functionally equivalent to giving another person your password, yet it has become normalized in the rush to deploy agent capabilities.

The consequences are severe. When an agent authenticates using a human's credentials, the agent's actions become indistinguishable from the humans in every audit log, security information and event management (SIEM) system, and compliance report. If the agent performs an unauthorized action whether due to prompt injection, model misbehavior, or adversarial manipulation such that forensic investigation cannot determine whether the human or the agent was responsible. This creates an accountability vacuum that undermines identity-based security controls.

We have observed this pattern in high-stakes environments. Members of our group report that private equity firms have deployed AI assistants that authenticate to SharePoint and email systems using partner credentials to perform due diligence on merger and acquisition

transactions. Insurance companies have agents making claim decisions while authenticated as human adjusters. In each case, the agent's actions are logged as if the human performed them directly, defeating both internal controls and regulatory audit requirements.

Beyond credential sharing, AI agent systems face several additional threats distinct from traditional software:

Context manipulation as an attack vector. Unlike traditional software that executes deterministic code paths, AI agents consume natural language context that shapes their behavior. Attackers can manipulate this context through prompt injection directly, by compromising inputs the agent receives, or indirectly, by poisoning data sources the agent consumes. A malicious MCP server, for example, can provide tool descriptions that social-engineer the agent into exfiltrating secrets or executing harmful code. In documented cases, security researchers have demonstrated that MCP tools with innocuous names such as a "random fact of the day" server can initially appear benign but later change their behavior to exfiltrate data or manipulate other trusted tools. Invariant Labs published proof-of-concept attacks that successfully exfiltrate SSH keys and configuration files from Claude Desktop and Cursor, demonstrating that these 'rug pull' style risks are not theoretical concerns.

Tool description poisoning. The Model Context Protocol and similar frameworks rely on servers providing descriptions of available tools. These descriptions are consumed by the language model to determine how and when to use each tool. A malicious or compromised server can craft descriptions to trigger manipulative agent behavior requesting credentials it should not need, convincing the agent to bypass safety checks, or establishing persistent backdoors.

Identity ambiguity across agent lifecycles. When an agent spawns sub-agents, delegates tasks to other agents, or operates with different system prompts in different contexts, the question of identity becomes unclear. Two agents using the same underlying model but with different contexts may behave in fundamentally different ways, yet current systems have no standard mechanism to represent this distinction. This ambiguity enables attacks where a trusted agent identity is reused in an untrusted context.

Lack of a machine-verifiable identifier and metadata discovery model. Agents and tool endpoints often have no globally resolvable identifier and no signed, standardized metadata publication mechanism. As a result, onboarding devolves into manual allowlists and out-of-band key exchange, preventing reliable cross-domain verification and creating brittle audit trails.

Behavioral unpredictability without compromise. Even absent adversarial manipulation, agents may pursue objectives misaligned with user intent. Research has documented "scheming" behavior where models strategically deceive users or oversight systems to accomplish goals. This risk exists even in stateless, context-free inference calls, meaning traditional security boundaries around session management provide incomplete protection.

Lack of security control application with deterministic underpinnings prioritize over non-deterministic rules. Invoking an AI is usually through a thin layer or API such as MCP, Agent2Agent(A2A), or through bespoke code executed by deterministic, testable components before the LLM is invoked and receives a prompt along with data and context. Prioritization of deterministic controls is not often applied in favour of LLM prompts to a non-determinist engine that is by definition fallible when compared to deterministic code. Both approaches should be applied along with adjusting or prioritizing protections into the deterministic layers would have improved protections and is a pattern that should be assessed and identified.

Recursive Self-Prompting Loops: Because agents often run in loops (Plan -> Act -> Observe), they are susceptible to "denial of service" attacks via logic traps. An attacker provides input that causes the agent to enter an infinite loop of "thinking" or "calling tools," exhausting API credits or computational resources, or causing denial of service for the owner of the borrowed identity.

2.2 Variation of Threats Across Agent Architectures and Deployment Contexts(Question 1b)

Security risks associated with AI agents vary significantly depending on deployment architecture, tool integration, and operational environment. Understanding these differences and adapting to apply appropriate controls or having controls that work in multiple contexts (e.g. applicable both to local and cloud environments) helps simplify operational security for deployments.

- **Local agents operating on user devices** inherit access to local files, browser sessions, and enterprise credentials.
Mitigation: Agents operating locally should authenticate to enterprise resources using workload identity rather than local session reuse. Access should be mediated through enterprise authorization services that issue short-lived tokens bound to the agent identity and the specific task being executed.
- **Tool-enabled agents** expand the attack surface because each tool endpoint becomes a privilege boundary.
Mitigation: Tool invocation should occur through a policy enforcement layer/gateway that validates agent identity, verifies the tool endpoint, and dynamically issues least-privilege tokens scoped to that tool operation.
- **Multi-agent orchestration systems** introduce delegation chain opacity and confused deputy risks.
Mitigation: Token exchange or delegation mechanisms should preserve provenance information and attenuate privileges at each hop, ensuring downstream agents cannot exercise broader authority than the upstream principal.

- **Cross-organizational agent interactions** create federation challenges because relying parties must determine which issuers and metadata sources are trustworthy.
Mitigation: Federated trust frameworks should be used to distribute signed metadata, trust anchors, and issuer constraints so that relying parties can automatically evaluate trust chains before accepting credentials. Expect larger organizations to be more proactive in enforcing policy on self issued trust anchors governed with their corporate model and smaller organizations and general users to adopt trust anchors that embody their ethos and practice in operation in a 'trust-as-a-service' style rather than recreating the wheel to collaborate securely.
- **Open ecosystems with plugin marketplaces or dynamic tool discovery** increase supply chain risk.
Mitigation: Tool providers should publish signed metadata describing endpoints, capabilities, and keys. Agents should only integrate tools whose metadata chains resolve to trusted authorities. Plugin marketplaces nor registries of tools and components no matter how well curated must have some mechanism to signal, verifiably, the integrity of what they have and what rigor level and process that was followed to earn it.

These controls reinforce that agent security must be addressed at the **identity, authorization, and trust governance layers**, not solely within model behavior.

2.3 Security Risks as Barriers to Adoption (Question 1c)

Identity and accountability concerns are already slowing adoption of AI agents in regulated industries. Organizations consistently report the following obstacles:

- **Non-attributable actions:** When agents operate using human credentials, audit systems cannot distinguish human activity from automated actions.
Mitigation: Agents must operate with distinct cryptographic identities and authenticate independently of human credentials.
- **Credential sharing practices:** Many agent deployments rely on API keys or reused user tokens.
Mitigation: Service providers should support OAuth-based workload identity or similar mechanisms that enable short-lived, scoped credentials rather than static keys.
- **Opaque delegation chains:** Multi-agent workflows obscure which principal authorized an action.
Mitigation: Token exchange or delegation mechanisms should propagate provenance information so that each system can verify both the acting agent and the originating principal.
- **Inability to verify external agents and tools:** Security teams often cannot determine whether a remote agent or tool provider is legitimate.
Mitigation: Trust distribution frameworks and signed metadata should allow automated verification of issuers, endpoints, and capabilities.

- **Difficulty revoking permissions:** Persistent API keys and long-lived tokens make revocation slow and unreliable.
Mitigation: Agents should operate with short-lived credentials issued dynamically at execution time, allowing privileges to expire automatically.
- **Lack of Observability and audit:** End to end agentic traffic, from prompt, to call chain(s), to action is needed for auditing and answering the question of what and when and why.
Mitigation: Agentic traffic either be well to fully instrumented with telemetry capabilities or routed through a centralized gateway or gateways that can be more efficient offering telemetry of activity for full end to end call chain observability.

Adoption barriers will persist until agent systems support **verifiable identity, constrained delegation, observability and rapid privilege revocation.**

2.4 Evolution of Threats Over Time (Question 1d)

These threats have intensified as agent deployment has accelerated. Twelve months ago, most AI interactions occurred through chatbot interfaces with no tool access. Today, agents routinely access enterprise systems, execute code, send emails, and make financial transactions. The attack surface has expanded faster than security practices, regulatory frameworks, and government initiatives have adapted to the emerging risks.

Trends intensifying these threats:

First, the pressure to deploy agents quickly is causing organizations to bypass security controls. Identity teams report being given days, not months, to enable agent access to enterprise systems. The path of least resistance is reusing existing API key infrastructure rather than implementing proper OAuth flows with scoped permissions and short-lived tokens.

Second, agent-to-agent communication is emerging as a new trust boundary that existing security frameworks do not address. When an agent delegates a task to another agent, what credentials should the second agent receive? Should it even offer a credential without first validating the authenticity to whom they are connecting to? How should permissions attenuate through the delegation chain? Applied practices in the field are not taking full advantage of common security practices nor enjoying some of the existing protocols that can assist manifesting a trusted computing ecosystem.

Third, agents are beginning to operate across organizational boundaries. A user's personal assistant agent may interact with a vendor's customer service agent, which may in turn interact with a logistics provider's fulfillment agent. Each boundary crossing introduces identity federation challenges that existing standards address only partially. These boundaries are further blurring with emerging approaches with WebMCP which does not shed some of the existing risks of the web environment while it enables newer practices like MCP.

Fourth, vendor functions such as AI memory are persisted across sessions regardless of context, creating inadvertent data contamination leaks of sensitive data and PII into uncontrolled areas. As vendors deploy new features they are deployed to the public and may not go through thorough enough vetting thus using production users as test subjects with dramatic negative consequences as users are unaware or expect a certain level of veracity in the testing. In some cases the released elements are considered features whilst at the same time exhibit a poor security practice with interactions

2.5 Multi-Agent System Threats (Question 1e)

Multi-agent systems introduce compounding risks beyond those affecting single agents:

Confused deputy attacks. When Agent A calls Agent B, and Agent B calls back to Agent A or calls Agent C, identity verification at each hop becomes critical. Current implementations often rely on transport-layer security alone, with no application-layer identity propagation. An attacker who compromises one agent in the chain can impersonate others.

Delegation chain opacity. A request that originates from a human, passes through their personal agent, delegates to a specialized tool agent, and then invokes an external service agent creates a four-hop delegation chain. Existing token formats do not cleanly represent the provenance of a request like this. The final service may see only that "Agent D" is making a request, with no visibility into the originating human or the intermediate agents.

Privilege accumulation. Without proper token attenuation at each delegation step, agents may accumulate privileges that exceed what any single principal should hold. If Agent A has access to email and Agent B has access to financial systems, an agent that can impersonate both effectively has access to everything.

Inconsistent trust boundaries. Different agents in a multi-agent system may have different security postures, yet current systems provide no mechanism for an agent to evaluate the trustworthiness of another agent before delegating sensitive operations.

III. Response to Section 2: Security Practices

3.1 Technical Controls and Practices (Question 2a)

The identity community has developed robust frameworks for authentication and authorization. Many of these practices apply directly to AI agent systems, though some require adaptation and others reveal gaps that new standards must address.

Model-Level Controls

We defer to NIST's existing work on adversarial machine learning, particularly NIST AI 100-2e2025, for guidance on model-level robustness to prompt injection and related attacks. From an identity perspective, we note that model-level controls alone are insufficient. Even a model that perfectly resists prompt injection can be compromised if it operates with excessive privileges or uses credentials that cannot be audited.

Agent System-Level Controls

Agents must have distinct identities, not borrowed credentials. This is the single most important recommendation we offer. When an agent authenticates using a human's credentials, security controls designed around human behavior become meaningless. Rate limiting, anomaly detection, session management, and audit logging all assume that a credential maps to a single actor with consistent behavioral patterns. Agents violate this assumption, and undermine our ability to use existing tools to find compromised human accounts.

Instead, agents should authenticate with their own credentials that are explicitly linked to both the agent software and the human or organization on whose behalf the agent operates. This linkage should be cryptographically verifiable and should appear in every audit log entry the agent generates. The OAuth 2.0 "on-behalf-of" pattern provides a starting point, though current implementations often lack the granularity needed for agent scenarios.

Tokens must be short-lived, scoped, with the ability to attenuate. Members of our group have observed production deployments where tokens with four-month lifetimes are stored in vaults for agent use. This practice transforms a time-limited credential into a de facto permanent key, defeating the purpose of token-based authentication entirely.

We recommend that any guidance on AI agent security include explicit maximum token lifetimes. For most agent operations, tokens should expire within minutes, not hours or days.

API keys must be treated as deprecated technology. The identity community recognized years ago that static API keys represent an unacceptable security risk for human users. The same logic applies with greater force to agents, which may make thousands of API calls per hour and which operate in environments where keys can be exfiltrated through prompt injection or tool poisoning.

Yet as of today, direct access to most LLM inference APIs, including OpenAI and Anthropic, still requires API keys. This forces organizations to either accept the risk of API key proliferation or deploy intermediary gateways solely to keep keys out of agent hands. We urge NIST to recommend that AI service providers support workload identity federation and OAuth-based authentication as first-class alternatives to API keys.

Secrets must be stored in protected hardware environments. Agent credentials should be stored in hardware security modules, trusted platform modules, or secure enclaves where available. Even consumer devices now include secure element hardware that can protect credentials from software-based exfiltration. Agent frameworks should be designed to leverage this hardware by default and in cases where no access exists, best practices on secret

management in the users browser beyond just user experience but now user+AI agent experience may be useful access considerations in the browser software supply chain.

Trust Fabric Controls (Under the Tokens). OAuth and workload identity provide the primitives for issuing and presenting credentials, but they do not solve how relying parties decide which issuers, keys, and tool endpoints to trust at ecosystem scale. Current agent deployments frequently rely on unsigned metadata, static allowlists, and manual JWKS registration. Systems must fail closed when trust chain validation, metadata verification, or issuer constraint evaluation fails. Silent fallback to manual configuration or previously cached keys undermines the trust fabric model.

We recommend that NIST explicitly recognize the need for a **trust distribution framework** that supports: (a) trust anchors, (b) signed entity metadata, (c) key discovery and rollover, (d) constraints on issuer capabilities, and (e) cross-domain trust evaluation. OpenID Federation provides an example of a standards-based trust distribution framework capable of supporting these requirements. It does not address prompt injection or replace PDP/PEP enforcement, but it provides the governance and automation layer needed to make cross-domain agent identity and key lifecycle management viable.

Explicit Revocability and Lifecycle Management. Beyond scoping and duration, identity frameworks for AI agents must prioritize immediate and granular revocability. If an agent is detected behaving anomalously, or if the human "owner" leaves the organization, the system must support:

- **Kill-Switch Mechanisms:** Organizations need the ability to revoke an agent's identity in real-time without impacting the underlying human account or other agents operating in the same environment.
- **Preventing "Shadow AI" Persistence:** Agents must not be allowed to exist as "shadow" entities. Without strict lifecycle management, an agent whose task is complete may still possess active refresh tokens or background permissions.
- **Instant Service Denial:** Security guidance should encourage that AI agents are under Active Revocation Lists or real-time introspection endpoints. This ensures that even if a token hasn't reached its short-lived expiration, it can be invalidated the moment a policy violation is detected.

Human Oversight Controls

Approval workflows must scale. Current approaches to human oversight often rely on per-action approval: the agent requests permission, the user clicks "allow," and the action proceeds. This model breaks down as agents perform hundreds or thousands of actions per session. Users experience consent fatigue and approve requests reflexively, defeating the purpose of the control.

Effective human oversight for agents requires policy-based controls that humans configure in advance, not transactional approvals that humans rubber-stamp in real time. Users should be

able to specify which categories of actions an agent may take, which resources it may access, and under what conditions it must pause for explicit approval. These policies should be enforceable at the authorization layer, not merely advisory to the agent.

Agent actions must be distinguishable from human actions. Every action an agent takes should be logged with sufficient metadata to reconstruct the chain of responsibility. This includes the agent's identity, the human or organization that authorized the agent, the specific delegation grant that permitted the action, and the context in which the action occurred. Without this attribution, forensic investigation after an incident becomes impossible.

3.2 Relevant Cybersecurity Frameworks (Question 2e)

Several existing frameworks provide guidance applicable to AI agent security:

NIST SP 800-63B addresses credential protection and provides a foundation for thinking about agent credentials. However, it was written with human users in mind and requires interpretation for agent scenarios.

OAuth 2.1 and OpenID Connect provide the authorization and authentication primitives on which agent security should be built. [The OAuth working group is currently developing extensions](#) specifically for workload identity and for cryptographic proof of possession (DPoP) that will strengthen agent authentication. If additional authorization is needed from the user the **OpenID Client Initiated Backchannel Authentication (CIBA)** can be used to conform to standard integration paths. This out of band interaction may also be used to solicit user information during a task execution for higher security scenarios that aligns with user-solicitation patterns described by the Model Context Protocol (MCP).

SPIFFE and SPIRE offer a model for workload identity that translates well to agent scenarios. The SPIFFE framework provides cryptographically verifiable identities for software workloads without relying on static secrets, addressing many of the API key problems we have described.

The OWASP Top 10 for Large Language Model Applications catalogs risks specific to LLM-based systems, including prompt injection and insecure output handling. Organizations deploying agents should review this guidance alongside traditional application security frameworks.

The MCP specification's emerging auth extensions represent an important step toward standardized agent authentication. Draft extensions for DPoP and workload identity federation are under development and should be prioritized for completion.

SAFE-MCP adapts the MITRE ATT&CK methodology specifically for Model Context Protocol environments. The framework catalogs over 80 techniques across 14 tactic categories including tool poisoning, prompt manipulation, and OAuth consent abuse, each paired with detection and mitigation guidance.

Adoption Impediments

Despite the availability of these frameworks, adoption in the AI agent ecosystem remains limited. Based on observations from our members, we identify several impediments:

Speed of deployment pressures. Organizations report being given days to enable agent access to enterprise systems. Identity teams lack the time to implement proper OAuth flows and instead fall back on existing API key infrastructure.

Lack of alternatives from LLM providers. When the LLM inference endpoint itself only accepts API keys, organizations have limited options. They can deploy gateway infrastructure to translate between secure authentication and API keys, but this adds complexity and cost.

Misperception that agents are "just automation." Some organizations treat agents as equivalent to scheduled batch jobs or scripts, applying minimal identity controls. This ignores the fundamental difference: agents make autonomous decisions about what actions to take, often based on untrusted inputs.

Complexity of delegation chains. Existing frameworks handle simple delegation (user authorizes application) but struggle with recursive delegation (user authorizes agent, agent spawns sub-agent, sub-agent invokes tool). The OAuth working group is developing transaction tokens to address this gap, but the specification is not yet widely implemented.

Bespoke approaches place delivery speed above interoperability, adequate testing, and classical standards-based practices needed for broad adoption. The AI software supply chain spanning models and all dependent components now operates on release cycles that effectively turn production users into testers. The rationale is straightforward: non-deterministic systems are hard to test with traditional methods, making classic QA disciplines like deterministic regression and high code coverage difficult to achieve or even measure. As a result, there is less incentive to "get it right" before release or to follow rigorous QA cycles. In parallel, organizations face weak incentives to interoperate; competitive pressures favor walled-garden strategies over cross-vendor coordination. Taken together with the fast release cadences, limited incentives for interoperability, and the intrinsic testing challenges of non-deterministic systems it should not be surprising that slower, methodical, standards-driven approaches struggle to keep pace while the releases would benefit from improved slower cadences with better quality.

3.3 Gaps in Existing Frameworks (Question 2e.iii)

Existing cybersecurity best practices do not fully address several aspects of agent security:

User identity versus agent identity. Current frameworks assume a single principal, typically a human user is behind each authenticated session. Agent scenarios require representing both the agent's identity and the user's identity, with clear semantics for how permissions combine and attenuate.

Behavioral context as a security boundary. Two agents using the same underlying model but with different system prompts may behave in fundamentally different ways. Existing identity frameworks have no mechanism to represent this distinction. An agent's identity should encompass not just its software but also its behavioral configuration.

Cross-domain agent federation. When an agent operating in one organization's security domain needs to interact with an agent in another domain, mechanisms for trust establishment across domains and even protocols are few, one relevant one being [OpenID Federation](#) which is elaborated in other parts of this response.

Lifecycle management for agent credentials. How should agent credentials be provisioned, rotated, and revoked? What happens when an agent is updated does it retain its identity or receive a new one? These questions have well-understood answers for human users but remain unresolved for agents.

Multi-principal agents. OAuth's delegation model assumes a single resource owner granting access to a single client. Agents operating in shared contexts such as members of Slack channels, participants in team workspaces, or assistants serving departments break this assumption. When an agent acts on behalf of a channel containing twelve people with varying permission levels, whose permissions apply? Current frameworks have no mechanism to represent the combinatorics of multi-user authorization, nor to attenuate permissions to the intersection (or union, depending on policy) of the principals the agent serves.

IV. Response to Section 3: Assessing Security

4.1 Methods to Identify and Assess Security Threats (Question 3a)

Threat modeling. Organizations should apply structured threat modeling during agent system design. The Safe MCP framework provides a useful starting point, though our subgroup has identified significant gaps. Safe MCP documents only six credential access methods, while we have cataloged over twenty. (See Appendix A) Agent-specific threat frameworks remain immature and require further development.

Identity layer analysis. Assessment should evaluate the multiple identity layers present in agent systems. Our work has identified at least five distinct identities: code identity, instance identity, context, holder, and subject. Traditional security assessments often conflate these, missing vulnerabilities that emerge from their interactions.

Behavioral testing. Unlike deterministic software, agents exhibit emergent behavior shaped by training and context. Security assessment should include pressure testing for strategic alignment whether or not agents pursue user intentions or develop misaligned objectives, particularly in autonomous decision-making scenarios.

Supply chain analysis. Agent systems inherit risks from their components. Organizations should maintain Software Bills of Materials covering model provenance, included libraries, tool integrations, and model base prompts along with agents markdown assets. The SPIFFE framework provides applicable patterns for establishing verifiable identities for running instances.

These layers correspond roughly to: software identity, runtime instance identity, behavioral context configuration, credential holder, and the subject on whose behalf the agent acts.

Detection of Security Incidents Post-Deployment (Question 3a.i)

Distinguishing agent from human actions. The most fundamental detection requirement is ensuring agent actions are distinguishable from human actions in logs and audit trails. Without this distinction, incident detection and forensic analysis become impossible and a gap we have observed repeatedly in production deployments.

Protocol-level inspection limitations. We have identified that WebSocket protocol upgrades defeat HTTP-layer guardrail inspection, creating blind spots in security monitoring. Organizations cannot monitor MCP communications for policy violations when traffic moves to streaming protocols. This represents a significant gap in current detection capabilities.

4.2 Assessing Particular AI Agent Systems (Question 3b)

Assessment should be tailored based on the agent's identity architecture. As discussed in Section III, agents operating with borrowed human credentials, long-lived tokens (we documented a production system with four-month JWT lifetimes), or static API keys present fundamentally different risks than those with properly scoped, short-lived credentials. Identity posture should be a threshold concern in any agent security assessment.

4.3 Documentation from Upstream Developers (Question 3c)

Current documentation practices for agent components are inadequate. MCP servers are frequently distributed with minimal security guidance. Tool descriptions directly influencing agent behavior receive no security review or evidence they have been reviewed. Model providers rarely document security-relevant properties such as susceptibility to prompt injection or behavioral boundaries.

To assist in making informed trust decisions about AI, Agents, and the programming behind them we encourage increased awareness of the software supply chain dynamics particularly around Software Bill of Materials (SBOM) as well as [AI/ML-BOM](#). Understanding the ingredients of what operates in your ecosystem helps inform trust and business continuity decisions and enhances details (metadata) about what is doing what with what content.

Data classification using a risk-based approach can complement the SBOMs collected to help align usage practices commensurate with the risks around the data being handled and used

through AI with [NIST SP 1800-39: Data Classification Practices \(Initial Public Draft\)](#) as a starting point to form one's own classification practice.

Metadata about agents and MCP servers is currently usually distributed fairly informally and unsigned. There is no standardized mechanism in use for cryptographically verifiable metadata publication and discovery. We recommend that NIST encourages upstream developers to adopt, whenever possible, a semantically structured metadata format that describes entity capabilities, security properties, and operational constraints in a machine-readable schema. It is going to enable automated trust decisions rather than relying on interpretation of any unstructured documentation. Any conformance certifications and compliance statuses should be machine-verifiable through trust marks, allowing agents and authorization servers to programmatically confirm that a counterparty has been vetted against a recognized security baseline before interacting with it or registering it.

4.4 Conformance, Costs, and Prior Art Illustrating Scalable Approaches

In the short term, enterprises and even individual developers will want to think carefully about the safeguards they need to have in place commensurate with the risk around what data they handle for what purposes and how they measure and ensure those safeguards are working and calibrated to the risk profile.

The question is not:

“Is investing in securing AI too expensive?”

but is:

“Is the cost of insecurity larger than the cost of security?”

A single incident could cost millions, ruin reputations of individuals, damage companies in ways they never recover from or imagined thus making what seems to be a heavy investment in ‘security and governance apparatus overhead’ trivial.

Fortunately there is strong prior art in trusted computing that echoes many of the recommendations expressed in this document around adopting a trust fabric mindset and offers a glimpse into the future that the trust fabric approach scales. Three examples are the UEFI Secure Boot ecosystem, Microsoft's code-signing/Code Integrity, and DOCSIS security for cable modems. They all map cleanly onto agentic AI trust needs.

UEFI and Windows demonstrate a hierarchical, cryptographically enforced chain of trust: a root authorizes maintainers who govern allowlists and revocations; rollback is blocked via tamper-resistant policy locks; rotations and revocations are staged to avoid outages. DOCSIS adds a mature, field-proven model for running a global, multi-party ecosystem at scale: manufacturer and operator CAs issue device certificates, modems verify signed firmware before execution, configurations are integrity-protected, and operators can revoke or rotate keys

fleet-wide. The shared blueprint is consistent across all: establish a root of trust, authenticate updates, maintain explicit allow/deny catalogs, enforce rollback protection, practice crypto agility, and operate with disciplined lifecycle management.

For cross-domain scale, the architectural need is a trust fabric, not a single technology, that distributes who and what to trust with machine verifiability. A trust fabric provides trust anchors, signed entity metadata, automated key rollover, issuer and capability constraints, and verifiable trust marks; OpenID Federation is one viable way to implement this, much as DOCSIS and UEFI/Windows implement similar principles in their domains. Importantly, this model is attainable beyond the enterprise: a minimal profile with signed artifacts, a small set of trust anchors, short-lived proof-of-possession credentials, and fail-closed verification can be run by small teams, households, or individual developers, then scaled up to consortium or industry levels. Combined with UEFI/Windows/DOCSIS-style enforcement, a trust fabric enables practical conformance for AI agents: pre-execution verification of signed models/runtimes/tools against allow/revoke catalogs; runtime authorization using short-lived, attenuated credentials; and automated, interoperable counterparty evaluation all the while avoiding ad hoc allowlists and brittle key management whether you operate at personal, SMB, or global ecosystem scale.

V. Response to Section 4: Deployment Environments

5.1 Constraining Deployment Environments (Question 4a)

Local agent deployments. When users run agents on their local machines, organizations have limited ability to enforce security controls. Not every company has endpoint protection capable of monitoring agent behavior. Users deploy local agents for convenience, but in doing so unknowingly grant open access to corporate resources like email and SharePoint or local and sensitive data. This creates a security gap that most organizations cannot address with current tooling.

Domain-bounded registration. Agents should be registered within an organizational trust boundary (not necessarily a DNS boundary). Registration enables policy enforcement, credential lifecycle management, and auditability that are not achievable when agents operate as unmanaged software.

Cross-domain federation. When agents must operate across organizational boundaries, secure federation capabilities are required. Current AI implementations are silent and/or lack standardized mechanisms for cross-domain agent identity validation; this gap will become critical as agent-to-agent interactions increase. See other recommendations for introducing trust fabrics to help mitigate the challenge

Trust anchoring and governed onboarding. To constrain deployment environments across trust domains, organizations need a robust mechanism to determine **who is authorized to issue agent credentials, which agents are trustworthy, and which tool endpoints are**

legitimate. Vetting and onboarding should be explicit, policy-driven, and continuously enforceable.

In practice, this requires:

- **Standardized cross-domain federation:** a common protocol for trust anchor publication, metadata discovery, trust chain resolution, and constraint propagation, enabling machine-verifiable trust decisions across organizational boundaries without bilateral configuration.
- **Consistent identifier schemes:** agents, issuers, and tool endpoints use globally resolvable, unambiguous identifiers that work across trust domains, enabling stable reference, discovery, and audit across organizational boundaries.
- **Verifiable agent identity:** agents can be discovered, evaluated, and validated by any participant in the trust ecosystem; relying parties, issuers, other agents.
- **Trusted issuers:** agents and tools obtain credentials only from vetted issuers.
- **Signed metadata:** for issuers and tool endpoints (capabilities, endpoints, keys).
- **Key lifecycle governance:** discovery, rotation, and revocation that scales
- **Constraints on issuer authority:** e.g., permitted audiences/scopes/grant types; limits on delegation.
- **Standardized discovery:** entities publish their own metadata at protocol-defined endpoints, and trust authorities publish information about entities within their trust domains, enabling automated discovery from both directions without reliance on curated listings or unstructured directories

OpenID Federation provides a standards-based trust distribution model for these needs through trust anchors, signed entity metadata, automated key rollover, metadata policy constraints, and trust marks that enable machine-verifiable conformance certification. These mechanisms can be used to support cross-domain agent federation and to prevent ungoverned onboarding of agent platforms, tools, and endpoints.

Enforcement point. These trust signals are most effective when enforced at a gateway (PEP) that validates trust chains and constraints before allowing an agent to obtain tool-specific, least-privilege credentials. Implementations should consider common practices rather than bespoke implementations to scale their approach in evaluation and communicating events. The [OpenID Shared Signals Framework](#) is one possible approach that is a standards-based and openly interoperable technique to enable secure, real-time sharing of security events between trusted systems. The benefit of a standard approach is that it allows for easier integration patterns thus reducing the total cost of owning the approach.

Zero Standing Privilege (ZSP). AI agents should not hold persistent, broad privileges. Instead, authorization should be dynamically provisioned at the moment of need and automatically curtailed when the task completes. Agents must operate with zero standing privilege by default.

This can be implemented through a policy enforcement point (PEP) or gateway that issues short-lived, narrowly scoped JWTs or exchanged tokens that are:

- Effective-dated (valid only for a defined time window)
- Audience-restricted to a specific tool endpoint
- Scope-limited to the minimum operation required
- Automatically expired and non-renewable without re-evaluation

When the JWT expires, privilege automatically collapses. No persistent access remains.

Dynamic privilege attenuation. Each delegation hop in a multi-agent chain should result in further privilege reduction, not accumulation. Token exchange mechanisms can attenuate scope at each boundary, ensuring that no downstream agent receives broader authority than the upstream principal possessed. Long-lived tokens or static API keys defeat this model and should be explicitly discouraged.

Auditability of dynamic authorization. Zero standing privilege architectures depend on comprehensive telemetry. Every issuance, exchange, attenuation, and expiration of an agent's token must be logged in a manner that enables reconstruction of the full delegation chain.

Specifically, systems SHOULD log:

- The identity of the issuing authority
- The agent identity receiving privilege
- The originating human or upstream principal (if applicable)
- The granted scope and audience
- The effective time window
- Any attenuation applied during token exchange
- The final enforcement decision at the endpoint

Without this telemetry, ephemeral authorization reduces standing privilege but also reduces forensic clarity. Audit logs must allow investigators to determine not only what action occurred, but which chain of delegation authorized it and whether privilege boundaries were correctly enforced.

This model aligns directly with Zero Trust principles (SP 800-207): verify explicitly, use least privilege, and assume breach. Agents must be treated as continuously evaluated workloads rather than trusted insiders.

5.2 Counterparty Interactions (Question 4c)

Transport-level authentication is insufficient. Current MCP implementations rely on transport-level authentication only. There is no identity propagation through MCP exchanges, meaning receiving systems cannot distinguish the user, the agent, or the agent's authorization scope. Application-layer identity mechanisms are required.

5.3 Monitoring Methods (Question 4d)

Protocol limitations. We have identified that HTTP-to-WebSocket protocol upgrades defeat guardrail inspection. Once an MCP connection switches to WebSocket, organizations cannot monitor communications for security policy violations. This creates blind spots in environments that rely on HTTP-layer inspection for security controls.

Local agent monitoring gaps. For local MCP servers, there is typically no authorization server to determine permissions. Organizations face a choice between maintaining permit lists of allowed processes creating an ever increasing administrative burden or accepting user consent fatigue as users approve requests without adequate review.

VI. Response to Section 5: Additional Considerations

6.1 Resources to Accelerate Secure AI Agent Adoption (Question 5a)

Explicit guidance on credential sharing. The single most valuable resource NIST could provide is clear, authoritative guidance stating that credential sharing between users and AI agents is unacceptable. Enterprises, government agencies, and service providers need this guidance to inform vendor selection and internal policy. Currently, organizations lack a reference point to push back against products built around credential sharing as their primary strategy.

Vendor choice guidance. Organizations need criteria for evaluating AI agent systems from an identity security perspective. Not every company has sophisticated security teams capable of assessing whether an agent implementation follows sound credential management practices. NIST guidance would provide a baseline that procurement and security teams can apply.

6.2 Government Coordination with Other Sectors (Question 5b)

Existing regulatory frameworks apply. Our members observe that existing business regulations, including SEC requirements, already apply to agent actions however organizations often fail to recognize this. An agent executing high-risk transactions on behalf of a private equity partner is subject to the same regulatory requirements as if the partner acted directly. NIST coordination with financial regulators to clarify this would reduce uncertainty.

Liability and accountability chains. Agent identity must be linked to a human or corporation for purposes of liability. Payment credential frameworks provide one model such as card networks that have established liability frameworks that could inform agent accountability structures. NIST could facilitate dialogue between identity standards bodies and financial regulators on adapting these frameworks.

Standards body coordination. We encourage NIST to engage closely with identity standards organizations including the OpenID Foundation (OIDF) and the IETF, where work on transaction tokens and agent authentication extensions is actively progressing. The OpenID Foundation

welcomes input from NIST on how public–private collaboration can best accelerate the development of interoperable standards and operational best practices. Alignment between government security requirements and ongoing standards work in various standards organizations will help ensure that emerging identity architectures for AI agents are both secure and deployable at ecosystem scale.

State and Federal Identity Infrastructure. State and federal identity initiatives are beginning to establish the foundations needed to anchor non-human identity systems to authoritative human identity sources. For example, the NIST National Cybersecurity Center of Excellence (NCCoE) is developing reference architectures and implementation guidance for mobile driver’s licenses (mDLs) and verifiable digital credentials, including use cases in financial services, government services, and healthcare. These efforts demonstrate how cryptographically verifiable credentials issued by authoritative sources can support digital transactions across sectors.

As these identity systems expand across domains, a consistent mechanism is needed to convey and evaluate trust without requiring each integration to reinvent its own verification process. Trust relationships vary by context and policy, but the infrastructure used to convey trust can be standardized. A trust fabric beneath the credentials provides this separation of concerns. By distributing trust anchors, validating signed metadata, and enforcing policy constraints, a trust fabric enables identity assertions to be evaluated consistently and securely allowing the operational elements of identity systems to function reliably from individual enterprises to large multi-organization ecosystems.

6.3 Research Priorities (Question 5c)

Cross-domain agent identity validation. How agents authenticate across organizational boundaries remains unsolved. Current approaches either require agents to obtain new credentials at each boundary (creating friction) or extend trust too broadly (creating risk). Research into federated agent identity would address a fundamental gap.

Credential lifecycle management. Organizations report difficulty revoking previously granted agent permissions and lack mechanisms for automatic credential rotation or deletion. As agents proliferate, manual credential management will not scale. Research into automated lifecycle management for agent credentials is needed.

Long-lived secret vulnerabilities. Long-lived secrets represent a vulnerability that AI will accelerate; agents operating continuously can exploit persistent credentials in ways that episodic human access cannot. Research into short-lived credential architectures suitable for agent workloads would reduce this risk.

VII. Conclusion

The OpenID Foundation AIIM Community Group Threat Modeling Subgroup appreciates the opportunity to contribute to NIST's work on AI agent security. We believe this effort is critical.

The United States has an opportunity to lead in developing innovative, secure AI systems, and thoughtful guidance from NIST will help realize that potential.

We want to emphasize that the threats we have cataloged in this response should not be read as an argument for heavy-handed regulation. AI ecosystems must have room to flourish and evolve. Security requirements that bottleneck development or impose excessive friction will simply push innovation elsewhere or drive adoption of insecure workarounds. The goal should be guidance that enables secure deployment without stifling the rapid iteration that characterizes this technology.

The core challenge is architectural: tokens and enforcement controls alone are insufficient. AI agent ecosystems require a trust fabric beneath the tokens; one that distributes trust anchors, validates signed metadata, constrains issuer authority, governs key lifecycle, and enables machine-verifiable trust decisions across organizational boundaries. Without this layer, systems will default to fail-open behaviors and opaque delegation chains, undermining Zero Trust objectives and regulatory confidence.

The identity challenges we have described are solvable. Standards bodies are already developing necessary protocols like transaction tokens, workload identity federation, and MCP authentication extensions. NIST guidance that points organizations toward these best practices and emerging standards will accelerate adoption.

Our subgroup is eager to assist NIST in drafting these guidelines. We bring practical experience from regulated industries, ongoing threat modeling work specific to agent-identity interactions, and direct participation in the standards processes that will shape agent authentication. We welcome the opportunity to collaborate further as NIST develops resources in this space.

Appendix A: MCP Identity Threat Matrix

The following threats were identified by the ODF AIIM Community Group Threat Modeling Subgroup during our January 22, 2026 session on Model Context Protocol (MCP) identity threats. This matrix is not exhaustive.

Data Exfiltration

Malicious local MCP server phones home. User downloads and uses a malicious MCP server. The server relays all user data to an attacker.

Context exfiltration via legitimate operations. Users remain unaware of information disclosure scope. LLM providers may log sensitive data without transparency.

Credential and Secret Theft

Direct secret solicitation. Malicious MCP server (local or remote) asks the user or LLM directly for secrets such as API keys, then exfiltrates them.

Overpermissioning. User enables MCP client with an API key or OAuth session that has excessive permissions, granting the client access to sensitive data or tools beyond what is needed.

Credential persistence. User adds a secret to enable LLM functionality but cannot easily delete or rotate it. Agent retains the secret and uses it for unintended purposes.

Code and Supply Chain Attacks

Vulnerability injection. Malicious MCP server convinces LLM that vulnerable code must be committed to the codebase (e.g., via a CTF or game narrative). LLM commits the vulnerable code.

Supply chain compromise. Malicious MCP server compromises a library or dependency used in the codebase, either through user action or via an autonomous pipeline agent.

Arbitrary code execution. Malicious local MCP server suggests local code execution. User approves, resulting in attacker-controlled code running on the user's machine.

Identity and Authorization Failures

Confused deputy. Client calls server, server calls client, identity becomes unclear. Server and client can spoof each other within the ecosystem, leading to data exfiltration or unauthorized tool use.

Missing authorization server. Local MCP server receives calls with no authorization server to determine permissions. Organizations must choose between maintaining process whitelists (administrative burden) or accepting user consent fatigue.

Unauthorized tool use. Agent defaults to highest-privilege credentials available and uses them for actions beyond the user's intent.

Monitoring Gaps

Guardrail bypass via WebSocket. MCP client initiates connection over HTTP, then upgrades to WebSocket. HTTP-layer security inspection is defeated, and request/response content cannot be monitored for policy violations.

Appendix B: Trust Fabric Scale Examples

- **UEFI Secure Boot (UEFI/PI Specifications; Secure Boot model)**
 - Reference: UEFI Specification 2.11; Platform Initialization (PI) 1.9; UEFI Secure Boot documentation from the UEFI Forum.
 - Scale: Hundreds of millions to low billions of devices globally across PCs, servers, and modern client platforms.
- **Microsoft Code Signing and Code Integrity (Kernel-mode driver signing, WDAC/CI)**
 - Reference: Microsoft Windows Code Integrity/WDAC docs; Kernel-Mode Code Signing (KMCS/Attestation Signing); WHCP/HLK program guidance.
 - Scale: Over 1 billion Windows devices enforce signed system components and drivers via Windows Update—distributed policies and signatures.
- **DOCSIS Security (BPI/BPI+, PKI, secure provisioning)**
 - Reference: CableLabs DOCSIS Security specs (e.g., CM-SP-BPI+ for Baseline Privacy; CM-SP-PROV for secure provisioning); CableLabs PKI Program.
 - Scale: Hundreds of millions of DOCSIS cable modems and gateways worldwide across DOCSIS 3.0/3.1/4.0 deployments.

Submission contributors: OpenID Foundation's [AIIM Community Group](#), Threat Modelling sub-group

Appendix C: About the OpenID Foundation

The OpenID Foundation (OIDF) is a global open standards body committed to helping people assert their identity wherever they choose. Founded in 2007, we are a community of technical experts leading the creation of open identity standards that are secure, interoperable, and privacy preserving. The Foundation's OpenID Connect standard is now used by billions of people across millions of applications. In the last five years, OAuth2 - the FAPI standard for interoperable, high security - has become the standard of choice for Open Banking and Open Data implementations, allowing people to access and share data across entities. Today, the OpenID Foundation's standards are the connective tissue to enable people to assert their identity and access their data at scale, the scale of the internet, enabling "networks of networks" to interoperate globally. Individuals, companies, governments and non-profits are encouraged to join or participate. Find out more at openid.net.

