Response to NCCoE Concept Paper

Accelerating the Adoption of Software and
AI Agent Identity and Authorization

AI-Identity@nist.gov

Comment Period: February 5 – April 2, 2026

ATTESTED INTELLIGENCE

Jack Brennan

admin@attestedintelligence.com

USPTO Patent Application No. 19/433,835
USPTO Trademark Serial No. 99677085
Illinois File No. 17233815

March 4, 2026

AttestedIntelligence.com

TABLE OF CONTENTS

Key Terms

Term	Definition
Attested Governance Artifact (AGA)	A sealed, cryptographically signed policy object encoding agent identity binding, authorized behavior, enforcement parameters, and measurement cadence
Subject Identifier	Cryptographic binding comprising hashes of the agent's normalized bytes and canonicalized metadata
Portal (Sentinel)	Runtime enforcement boundary that intercepts agent interactions and enforces sealed artifact constraints
Continuity Chain	Append-only sequence of signed events linked by structural metadata hashes, providing tamper-evident history
Evidence Bundle	Portable verification package containing artifact, receipts, Merkle proofs, and checkpoint references
Sealed Hash	Immutable reference value representing the agent's attested known-good state

INTRODUCTION

Attested Intelligence Holdings LLC submits this response to the NCCoE concept paper on software and AI agent identity and authorization. We are a security engineering firm building cryptographic governance infrastructure for autonomous systems in adversarial and disconnected environments. Our patent-pending architecture (USPTO Application No. 19/433,835, 20 claims) addresses the exact class of identity, authorization, and non-repudiation challenges this concept paper identifies.

Our core technology, **Attested Governance Artifacts (AGA)**, generates sealed, cryptographically signed policy objects that bind agent identity to authorized behavior and enforce that binding continuously at runtime. A portal execution boundary intercepts all agent interactions with external resources, measures agent state against a sealed reference, and generates signed receipts appended to a tamper-evident continuity chain. In NIST AI RMF terms, the AGA framework operationalizes the Measure function (continuous runtime integrity verification) and the Manage function (autonomous enforcement upon policy violation). The architecture has a working reference implementation with a validated cryptographic core (Ed25519 signatures, SHA-256/BLAKE2b hashing, HKDF-SHA256 key derivation, RFC 8785 JSON canonicalization) and demonstration applications for SCADA process enforcement and autonomous vehicle command governance.

We are interested in serving as a technology collaborator for this NCCoE project. The sections below respond to each question category in the concept paper and describe how the AGA framework addresses the identity and authorization challenges posed. We conclude with a proposed laboratory demonstration scenario.

NCCoE Question Category	AGA Response	Section
1. General / Use Cases	SCADA, autonomous vehicle, and MCP server deployments	Section 1
2. Identification	Subject Identifier + non-biometric cryptographic identity	Section 2
3. Authentication	Artifact signature verification, TTL-based re-attestation	Section 3
4. Authorization	Portal as zero-trust PEP, sealed policy constraints	Section 4
5. Auditing / Non-Repudiation	Continuity chain, Merkle anchoring, evidence bundles	Section 5
6. Prompt Injection	Behavioral drift detection, phantom execution forensics	Section 6

1. GENERAL: USE CASES AND AGENTIC ARCHITECTURE CHALLENGES

Agentic architectures differ from traditional microservices in a fundamental way: a microservice follows imperative logic along deterministic code paths. An AI agent follows probabilistic reasoning, making autonomous decisions about which tools to invoke, which data to access, and how to sequence operations. The set of actions an agent may take is not fully enumerable at deployment time. NIST SP 800-204 (Security Strategies for Microservices) provides the foundation for service mesh security, but its controls assume that service behavior is predictable from the code. Standard service mesh architectures (Istio, SPIRE, Envoy) secure the communication channel. The AGA framework secures the intent within that channel, governing what the agent decides to do with the access it has been granted.

The core challenge: an agent may be properly authenticated and initially authorized, yet produce unauthorized outcomes through unpredictable tool-chaining, context accumulation, or behavioral drift induced by adversarial inputs. Authentication and authorization at the point of entry are necessary but insufficient. Governance must be continuous.

Our engineering work has focused on three deployment contexts where these challenges are most consequential:

SCADA Process Enforcement. An AI agent monitoring or controlling industrial processes executes within a portal boundary. The sealed policy artifact specifies measurement cadence (100ms for control binary digest), authorized actuator connections, and enforcement actions upon drift detection. Critically, enforcement does not mean hard termination. For industrial control, the portal triggers a policy-defined safe state: locking valves to last-known-good positions, completing in-progress chemical transitions before severing control, or handing off to a manual backup system. The artifact distinguishes between "hard stop" (immediate termination for non-critical agents) and "safe-state transition" (graceful degradation for safety-critical systems).

Autonomous Vehicle Governance. Flight control software operates under a sealed artifact specifying return-to-home enforcement upon drift detection. If the control software is modified mid-flight, the portal severs attacker control and executes safe-landing protocols. Identity is bound to the vehicle's cryptographic key pair, not to any human operator.

MCP Server Integration. We are implementing a Model Context Protocol server where the AGA framework governs tool access and resource authorization for connected AI agents. The portal intercepts MCP tool invocations, verifies each call against the sealed policy artifact, and generates signed receipts for every interaction. This is direct implementation experience with the protocol the concept paper identifies as a key area of interest.

2. IDENTIFICATION

The concept paper asks how agents might be identified in enterprise architecture and what metadata is essential for an agent's identity. Our architecture answers this through two mechanisms: a **Subject Identifier** for binding an agent to its attested state, and a **non-biometric cryptographic identity model** for establishing persistent agent identity across interactions.

Subject Identifier (Claim 1(b)). Every governed agent receives a Subject Identifier comprising at least one cryptographic hash of the agent's normalized bytes (executable image, model weights, container digest) and at least one cryptographic hash of canonicalized metadata (version, author, configuration manifest, creation timestamp). This binding is computed at attestation time and sealed into the policy artifact. It answers the question "which specific version of which specific agent is this?" with cryptographic precision.

Non-Biometric Identity (Section M of our specification). Agent identity is derived from cryptographic key pairs rather than biometric traits. An agent's identity credential comprises a key pair bound to an append-only history of attestations stored in a continuity chain. Authority and rank are derived from the history of valid signatures, not from biological identity. This model enables AI agents, autonomous vehicles, and other non-human actors to hold cryptographic identity indistinguishable from human operators within the system.

This maps directly to the concept paper's discussion of non-human principals (NHPs). The Subject Identifier provides the *what* (which agent instance, which version, which configuration), while the cryptographic identity credential provides the *who* (which entity holds authority, what is its governance history). Both are machine-verifiable without network connectivity.

The concept paper asks whether agent identity metadata should be ephemeral or fixed. Our architecture provides both. The cryptographic key pair and continuity chain history constitute **fixed identity**: they persist across tasks and accumulate governance history over time. The sealed policy artifact constitutes an **ephemeral policy layer**: it is task-specific, has a defined TTL, and expires absent re-attestation. An agent's identity endures. Its authorization to act is temporary and scoped.

> **Integration Point: SPIFFE/SPIRE**
>
> Our identity model is complementary to SPIFFE/SPIRE rather than competitive with it. SPIRE handles node-to-workload identity ("this container is running on this node in this cluster"). The AGA Subject Identifier handles workload-to-intent governance ("this agent instance has been attested against this policy and its runtime state matches its sealed reference"). In a deployment, SPIRE issues the workload SVID while the AGA binds governance parameters to that identity. We have not yet implemented SPIFFE integration but the architectural interface is well-defined: the SVID provides transport-layer identity, the AGA provides governance-layer accountability.

3. AUTHENTICATION

The concept paper asks what constitutes strong authentication for an AI agent and how to handle key management. Our architecture addresses both.

Artifact Signature Verification (Claim 10). The portal maintains a pinned public key for the policy issuer and rejects any policy artifact whose signature does not verify against the pinned key. Before an agent is permitted to execute, the portal verifies the artifact signature (Ed25519 over canonical JSON serialization), confirms the current time falls within the artifact's effective period, and computes an initial integrity measurement. If any check fails, execution is blocked. This is authentication of the governance relationship, not just authentication of the agent itself.

Key Management Through Continuity Chains. Key issuance is recorded as a POLICY_ISSUANCE event in the continuity chain. Key rotation is handled by including both old and new public keys during a transition period defined by policy. Revocation is recorded as a REVOCATION event. The full key lifecycle is captured in the tamper-evident chain, providing auditable key provenance. This addresses the concept paper's question about issuance, update, and revocation for agent credentials.

Authentication in the AGA model is not a one-time event. The sealed artifact has a time-to-live (TTL) value (Claim 6). When the TTL expires, the agent must re-attest to continue operating. This creates a **continuous authentication** model: the agent's authority to act is not permanent but periodically renewed, with each renewal producing a new signed artifact and a corresponding chain event.

Relationship to OAuth 2.0. The concept paper identifies OAuth 2.0 as the primary authorization mechanism integrated into MCP. The AGA framework is complementary: the sealed policy artifact can serve as the authorization context bound to an OAuth access token. Even if a token is compromised, the portal enforces the sealed artifact's constraints independently. A stolen token without a valid, signature-verified artifact grants no execution authority. The artifact provides the governance layer that OAuth's token-based delegation model does not address on its own.

Algorithm Agility. The current reference implementation uses Ed25519 for signature performance (critical for the sub-10ms enforcement target in high-cadence measurement scenarios). The architecture is algorithm-agile by design: the artifact metadata specifies the signature algorithm, and the portal verifies using the algorithm indicated. As NIST's post-quantum standards move into federal procurement requirements, the framework is ready to adopt ML-DSA (Dilithium) or SLH-DSA (SPHINCS+) as drop-in replacements without architectural changes.

4. AUTHORIZATION

This is the area of strongest alignment between our architecture and the concept paper's objectives. The concept paper asks how zero-trust principles can be applied to agent authorization, how to establish least privilege, and how agents

prove authority to perform specific actions. The AGA framework provides concrete mechanisms for each.

The Portal as Policy Enforcement Point. In NIST SP 800-207 (Zero Trust Architecture) terms, the portal operates as a Policy Enforcement Point (PEP) and the sealed policy artifact serves as the Policy Decision Point (PDP) payload. The agent cannot bypass the portal to reach protected resources directly. Every tool invocation, API call, actuator command, and data access passes through the portal, which evaluates it against the sealed artifact's enforcement parameters. For high-consequence deployments (SCADA, autonomous vehicles), the portal itself should execute within a Trusted Execution Environment (TEE) such as Intel SGX or ARM TrustZone, providing hardware-level isolation for the governance logic. Our specification includes TEE attestation quotes as a measurement embodiment (Section E, Embodiment 6).

Least Privilege Through Sealed Constraints. The policy artifact functions as a **cryptographic mandate**: it specifies exactly which resources the agent is authorized to access, at what measurement cadence its state must be verified, and what enforcement actions apply upon policy violation. These constraints are sealed at attestation time and cannot be expanded at runtime. The agent cannot grant itself additional privileges, because the artifact's cryptographic signature would be invalidated by any modification.

Fail-Closed Semantics. If the artifact cannot be parsed, if the signature does not verify, if the effective period has expired, or if the initial measurement does not match the sealed reference, the agent is not permitted to execute. The default state is denial. This inverts the traditional model where agents operate by default and controls are applied after the fact.

Continuous Authorization via Runtime Measurement. Authorization is not checked once at entry. The portal computes a cryptographic integrity value representing the agent's runtime state at the measurement cadence specified in the artifact (Claim 1(f)). Each measurement is compared to the sealed reference. A match means the agent continues operating. A mismatch triggers enforcement. This implements zero-trust continuous verification at the agent governance layer.

Dynamic Policy Response. The concept paper asks whether authorization policies can be dynamically updated when agent context changes. The AGA approach handles this through TTL-based re-attestation rather than runtime policy mutation. When context changes require new authorization parameters, the agent undergoes re-attestation, producing a new sealed artifact with updated constraints. This preserves the immutability guarantee while enabling policy evolution.

Recursive Delegation and Sub-Agent Governance. Multi-agent workflows create a recursive delegation problem: when a primary agent tasks a secondary agent, the secondary must operate under governance that is at least as restrictive as the primary's. The AGA architecture handles this through constrained sub-mandates. The portal issues a derived artifact to the secondary agent with a TTL no longer than the parent's remaining TTL and a scope that cannot exceed the parent's authorized resources. The secondary agent's genesis

event is cryptographically linked to the parent's continuity chain, ensuring that every action taken by the sub-agent traces back to the original human authorization. Scope can only diminish through delegation, never expand. This prevents privilege escalation across agent-to-agent boundaries while maintaining the unbroken chain of accountability the concept paper requires.

Delegation and Accountability. A human authority signs the policy artifact before the agent begins execution, creating a cryptographic chain of delegation. The artifact's issuer identifier records which authority authorized the agent's mandate. The continuity chain records the full sequence of attestation, enforcement, and receipt events. If an agent deletes a database or issues an unauthorized command, the receipt chain establishes: who signed the policy artifact (the authorizing human), what the artifact permitted (the sealed constraints), whether the agent operated within those constraints (the measurement receipts), and when it deviated (the drift detection event). The human who signed the artifact cannot credibly deny authorization, because the signature is cryptographically bound to the specific policy that governed the agent's actions.

5. AUDITING AND NON-REPUDIATION

The concept paper asks: "*How can we ensure that agents log their actions and intent in a tamper-proof and verifiable manner?*" This is the core problem our continuity chain architecture was designed to solve.

Tamper-Evident Continuity Chain (Claim 3). Every enforcement action, every measurement result, every attestation event is recorded as a signed receipt appended to an append-only continuity chain. The chain is initialized with a genesis event containing a protocol version identifier, a root fingerprint derived from the chain's cryptographic key pair, and a specification hash binding the chain to a specific rule set (Claim 17). For highest-assurance deployments (SCADA, defense), the genesis event's root key pair can be bound to a hardware security module (HSM) or TPM, ensuring that the chain's root of trust is anchored in tamper-resistant hardware rather than software alone. Each subsequent event contains an event type, identifier, sequence number, timestamp, payload, payload hash, and event signature computed over the complete event.

Structural Metadata Linking for Privacy-Preserving Verification. A major concern with comprehensive agent auditing is that logging everything may expose sensitive model outputs, proprietary business logic, or PII contained in agent interactions. Our architecture addresses this directly. The leaf hash for each event is computed using only structural metadata: schema version, protocol version, event type, event identifier, sequence number, timestamp, and previous leaf hash. Payload data is deliberately excluded. A third-party auditor can verify the complete integrity of the enforcement chain, confirm that every measurement was performed on schedule, and validate that every enforcement action was properly executed, all without ever seeing the contents of any agent interaction. Payload integrity is independently protected through event

signatures and content-addressable payload hashes, available only to parties with authorized access.

Merkle Checkpoint Anchoring (Claims 3(d-f)). The chain is periodically checkpointed by batching events into a Merkle tree, computing the Merkle root, and anchoring that root to an immutable append-only storage network. The system receives a transaction identifier as a checkpoint reference and records an ANCHOR_BATCH event in the chain. This prevents history rewriting even if the local database is compromised.

Non-Repudiation. Modification of any event's structural metadata invalidates its leaf hash, propagating forward through all subsequent events. An agent (or an attacker who has compromised the agent) cannot alter the enforcement record without detection. The signed receipt chain provides cryptographic non-repudiation: this specific agent, operating under this specific policy, took these specific actions, at these specific times. Each receipt binds back to the issuing authority through the artifact signature and the chain's delegation history.

Forward Secrecy. The continuity chain's key rotation strategy (recording rotation events as signed chain entries with overlapping validity periods) ensures that compromise of a current signing key does not enable retroactive forgery of earlier receipts. Historical chain segments remain verifiable against the keys that were active at the time of signing, as recorded in the chain's own key lifecycle events.

Offline Verification (Claim 9). Evidence bundles containing the sealed artifact, relevant signed receipts, Merkle inclusion proofs, and checkpoint references enable audit without network connectivity to the originating system. All verification steps except checkpoint anchor validation can be performed offline.

Resilience in Disconnected Environments. For National Security Systems and critical infrastructure deployments where persistent cloud connectivity is itself a security vulnerability, the AGA architecture's offline verification capability is not a secondary feature but its primary value proposition. The portal continues enforcing the cached policy artifact during connectivity loss (Claim 12), executing safe-state transitions upon TTL expiration rather than failing open. Evidence bundles use content-addressable hashing, making them self-contained and verifiable using only locally available cryptographic material. No callback to any external service is required to prove that governance was maintained throughout the disconnected period.

SSDF Alignment. The AGA architecture automates key requirements of NIST SP 800-218 (Secure Software Development Framework): specifically, verification of software integrity at runtime (PS.3) through continuous measurement against sealed references, and collection of forensic data during security incidents (RV.1) through the continuity chain receipt mechanism and phantom execution capture.

6. PROMPT INJECTION PREVENTION AND MITIGATION

The concept paper asks what controls help prevent prompt injection and what controls minimize impact after injection occurs. The AGA architecture addresses the mitigation side: containing damage and capturing forensic evidence when an injection attack succeeds.

Detection Through Behavioral Drift. Prompt injection changes agent behavior without modifying the agent's binary footprint. Traditional integrity measurement (file hashing, binary attestation) produces clean results even when the agent is fully compromised. The AGA portal detects the downstream effects: if injection causes the agent to invoke unauthorized tools, access unauthorized resources, or produce outputs inconsistent with its sealed policy, the resulting state divergence triggers enforcement.

Phantom Execution for Post-Injection Forensics (Claim 11). When injection is detected through behavioral drift, the portal can transition the agent to a sandboxed phantom environment rather than terminating it. All connections to protected resources are severed. The agent continues executing, believing it is operating normally, while all outputs are captured rather than delivered. The portal continues delivering inputs (including the injected prompts) to capture the full attack sequence. All attempted outputs and state changes are logged as forensic receipts. This transforms a prompt injection incident into an intelligence-gathering opportunity. Because the forensic receipts are signed and appended to the continuity chain, they can be packaged into evidence bundles and shared with other organizations to inform defenses against the same injection technique, enabling collective defense without exposing the victim's proprietary data (the structural metadata is shareable; the payload contents remain protected).

Scope Limitation. The AGA framework does not prevent prompt injection from occurring at the model level. It operates at the system level to contain the consequences. The sealed policy artifact defines what the agent is authorized to do; the portal enforces those boundaries regardless of what the agent's reasoning engine has been manipulated into attempting. Prevention requires model-level defenses. Containment and forensic capture are where the AGA architecture contributes. The enforcement overhead is minimal: Ed25519 signature verification and BLAKE2b hash computation are both optimized for high-throughput operation, making sub-second measurement cadences practical even in latency-sensitive deployments.

7. PROPOSED LABORATORY DEMONSTRATION

The NCCoE concept paper seeks to build a practical demonstration using commercially available technologies. We propose the following scenario, which could be implemented in the NCCoE laboratory to demonstrate how sealed governance artifacts address identity, authorization, and non-repudiation for AI agents.

Scenario: Governed AI Agent in a Simulated SCADA Environment

Setup. An AI agent is deployed to monitor and recommend control adjustments for a simulated industrial process. The agent has access to sensor data, control interfaces, and a reporting API through MCP tool connections.

Phase 1: Attestation and Identity Binding. The agent's complete state (model artifacts, tool configurations, dependency chain) is attested against a defined policy. A sealed policy artifact is generated, encoding the agent's Subject Identifier, authorized tool access, measurement cadence, TTL, and enforcement triggers. The artifact is signed by the issuing authority. The genesis event initializes the continuity chain.

Phase 2: Authorized Operation. The agent operates within the portal boundary, accessing sensor data and issuing control recommendations through authorized MCP tool calls. Each interaction passes through the portal, which verifies the agent's runtime state at the specified cadence. Signed receipts are generated for each measurement and appended to the chain. The demonstration shows continuous authorization in action.

Phase 3: Simulated Prompt Injection. An adversarial input is introduced that attempts to cause the agent to access an unauthorized control interface or issue a command outside its policy boundaries. The portal detects the behavioral divergence when the unauthorized tool call is intercepted. The portal transitions the agent to phantom execution: actuator connections are severed, but the agent continues operating in a sandboxed environment while forensic receipts capture the attack sequence.

Phase 3b: Mid-Session Revocation. An administrator pushes a REVOCATION event to the continuity chain, invalidating the agent's active policy artifact. On its next measurement cycle, the portal detects that the artifact has been revoked, fails the TTL check, and transitions the agent to a safe state. This demonstrates real-time authority termination for a running agent without requiring network-level kill switches.

Phase 4: Offline Audit. An evidence bundle is generated from the continuity chain, containing the sealed artifact, all signed receipts (including forensic receipts from the phantom execution phase), Merkle inclusion proofs, and the checkpoint reference. A separate verifier (simulating an air-gapped auditor) independently verifies artifact signatures, receipt signatures, and Merkle proofs without network connectivity to the original system.

What This Demonstrates. Identity binding through cryptographic attestation. Continuous authorization via runtime measurement. Fail-closed enforcement upon policy violation. Mid-session authority revocation. Forensic capture through phantom execution. Tamper-evident audit through continuity chain receipts. Offline verification through portable evidence bundles. Each capability maps directly to the concept paper's areas of interest.

Proposed Success Metrics

Interception Latency. Measure the overhead the portal boundary adds to MCP tool calls during continuous measurement. The Ed25519/BLAKE2b cryptographic stack is optimized for high-throughput signing and

verification. Target: governance overhead under 10ms per tool invocation for the measurement and receipt generation cycle.

Non-Repudiation Verification. An external verifier (not Attested Intelligence) independently validates the evidence bundle from the simulated compromise scenario, confirming artifact signatures, receipt chain integrity, and Merkle inclusion proofs using only the public key and bundle contents. No access to the originating system or proprietary tools.

Forensic Completeness. Confirm that the phantom execution phase captured the full sequence of unauthorized actions attempted by the compromised agent, with each action recorded as a signed forensic receipt in the continuity chain.

CONCLUSION

The identity and authorization challenges described in this concept paper require concrete technical solutions. The Attested Governance Artifact architecture provides one: sealed policy artifacts that bind agent identity to authorized behavior, a portal enforcement boundary that implements continuous zero-trust verification, signed receipts that provide cryptographic non-repudiation, and evidence bundles that enable portable offline audit. Attested Intelligence Holdings LLC is prepared to contribute reference code, technical expertise, and integration support to the NCCoE laboratory environment.

Respectfully submitted,

Jack Brennan
Attested Intelligence Holdings LLC
admin@attestedintelligence.com
AttestedIntelligence.com
March 4, 2026

USPTO Patent Application No. 19/433,835
USPTO Trademark Serial No. 99677085
Illinois File No. 17233815