FSB FINANCIAL STABILITY BOARD

Sound Practices for Responsible Adoption of Artificial Intelligence (AI): Consultation report

Response to Consultation

Erik P (independent response)

1. Do you agree with the benefits and risks of AI adoption by financial institutions described in this report? Are there any substantive benefits or risks not covered?

I am responding as an individual practitioner. My first professional work was assisting Sholom Rosen in the design of patent flows for Citibank's Electronic Monetary System (EMS). I chair the Consent Task Force of the Creator Assertions Working Group, and in January 2026 I submitted a public response to NIST's Request for Information on AI agent security (Docket NIST-2025-0035), proposing runtime enforcement infrastructure for agentic action.

The report's account of benefits and risks is excellent and thorough. However, one structural gap remains. The sound practices govern institutions and their oversight of AI. No practice governs the agentic action itself at runtime.

In Aristotelian terms, Moral AI ≠ Ethical AI. Covered in the report are theory/wisdom (theoria/phronesis), but left out is behavior (praxis), which is the only observable vector.

Banking solved this problem for money. A cash withdrawal at another bank's ATM proceeds only after independent authorities each verify their own condition, identity, account standing, interbank relationship, and regulatory compliance, at the moment of the transaction. A Citibank card and PIN entered at a Bank of America ATM are sufficient credentials for a $200 withdrawal on Saturday. Settlement on Monday follows on the strength of the record the transaction creates. The action is gated at execution, and the record allocates liability afterward.

The report's agent controls all operate before or after the action: model selection, testing, permission manifests, human oversight, accountability assigned once harm has occurred. Model-level safety training likewise shapes dispositions (moral) rather than enforcing limits at runtime (ethical praxis).

The report states that real-time human monitoring of agent decisions becomes impractical at scale. Where oversight cannot scale, governance must attach to the action: a machine-checkable verification that (1) identity, (2) provenance, (3) consent, (4) authorization, and (5) liability are valid at the moment of execution, producing a tamper-evident receipt sufficient to reconstruct the action and allocate liability.

My NIST response specifies this pattern as an Accountability-Gated Deployment profile, and I am glad to provide it as supplementary material.

Within that verification, two conditions remain ungoverned by any standards body. Consent, a revocable state that must be verifiable as current when an action relies on it. Liability, an attestation binding a responsible party to the action itself. Absent both, if consent is withdrawn, the agent may still act, and losses from autonomous actions resist timely allocation. At scale, both are financial-stability concerns.

2. **Are the sound practices sufficiently comprehensive and clear to enable financial institutions' responsible AI adoption?**

3. **Do the sound practices strike an appropriate balance between managing risks relating to all forms of AI, and addressing some of the risks relating to emerging and new complex forms of AI, such as GenAI and agentic AI?**

   The balance is appropriate for GenAI. For agentic AI, the report observes that real-time human monitoring of agent decisions becomes impractical at scale, then relies on human oversight and after-the-fact accountability to fill the space. My concern is that where oversight cannot scale, governance must attach to the action itself. Consent and liability, the two conditions ungoverned by any standards body, are sharpest here.

4. **Are the sound practices sufficiently flexible to accommodate and address newer types of AI and responsible adoption over time?**

5. **Do the case studies in this report sufficiently highlight how different types of financial institutions can benefit from responsible AI adoption? Are there additional case studies for inclusion in the report? If so, please provide such case studies, particularly for nonbanks.**

6. **Do the case studies in this report provide actionable insights for financial institutions in their responsible AI adoption?**

7. **Are the definitions in the glossary clear and aligned with industry sound practices, including recent developments in AI?**

   The glossary defines neither consent nor its verification. Suggested entry: "Consent: a revocable authorization by a person or institution for a specific use of data, funds, identity, or another protected interest, valid only while current. Its status must be verifiable at the moment an action relies on it." This aligns the glossary with consent vocabularies maturing in open standards bodies (CAWG, SMPTE).

8. **Are there any comments you would like to make on this report? Please fill in.**

   The consent gap in agentic finance is but one instance of a general problem. The same absence of point-of-action consent that lets an agent act outside a bank's authority lets a synthesized voice or likeness be used outside a human's authority (cf. EU AI Act, Article 50). These conditions should be governed as open, interoperable infrastructure, where gated action performs identically at all scales. The report rightly flags concentration risk,

and closed governance of consent and liability would reproduce that risk catastrophically at the trust layer.

Response to NIST Request for Information

Security Considerations for Artificial Intelligence Agents

Docket Number: NIST-2025-0035 | Federal Register 91 FR 698

Submitted by: Erik P [FULL NAME REMOVED]
Date: January 26, 2026
Contact: [REMOVED]
Affiliation: [REMOVED]
Submission: [REMOVED]

Executive Summary

AI agents can autonomously send messages, move funds, and control physical systems, yet there is no standardized, machine-checkable mechanism to determine who authorized an action and who is accountable when harm occurs. This submission proposes **Accountability-Gated Deployment (AGD)**, a runtime enforcement pattern based on one rule: **no credential, no action.** High-impact external actions execute only when a policy control verifies identity, authorization scope, accountability binding, and configuration integrity.

AGD is implemented by a **Policy Enforcement Point (PEP)** that sits between agent runtimes and external tools and enforces authorization at the moment of action execution. Authorization credentials are bound to an approved runtime configuration fingerprint (model identifier/weights, scaffold, tool manifest, deployment settings); any drift invalidates authorization.

If verification succeeds, the PEP issues a short-lived, scope-limited **Capability Token** for that request. Executed actions generate **tamper-evident Action Receipts** retained for forensic analysis, insurance claims, and legal accountability.

Adoption pathway: In safety-critical domains (automotive, aviation, industrial systems), insurance already enforces pre-deployment accountability through coverage conditions requiring attribution, configuration documentation, and verifiable controls, because insurers must be able to price risk and assign liability when incidents occur. AGD provides equivalent infrastructure for AI agents. The same artifacts support enterprise risk management, contractual requirements, and regulatory compliance.

See Figure 1 (Core Control Flow) and Figure 2 (Sequence Diagram) in Appendix B.

AGD Core Control Loop

```
None
Agent requests action → PEP intercepts → verifies credential + config +
scope → issues token OR denies → tool executes → receipt logged
```

Example: A wire transfer request is intercepted by the PEP. Financial actions above policy thresholds require elevated verification; if unmet, the request is denied and a signed denial receipt is generated. If approved, execution proceeds and a signed action receipt is logged.

Requested NIST Action: AGD-1 Control Pattern Profile

Requested deliverable: A short control-pattern profile (mini-spec) plus starter schemas for consistent implementation.

Publish an **AGD-1 Control Pattern Profile** defining:

1. **Credential & Revocation Model:** identity, scope, configuration binding, validity window, status checking
2. **Execution Enforcement & Audit Model:** PEP verification sequence, capability tokens, action and denial receipts
3. **Scope Taxonomy & Risk Classification:** starter action classes (financial, identity, communication, physical, data) and tier mapping

Optional companion profile: Accountability Attestation Profile describing minimum evidence for third parties (insurers, enterprise risk teams): responsible-party bindings, approved tool manifest, trust boundary summary, configuration fingerprint method, receipt retention policy, fail-policy settings.

NIST alignment:

- Fits AI RMF "Measure/Manage" as deploy-time control family
- Fits SP 800-53 (access control + audit + configuration management)

- Fits CAISI agent hijacking work: denial receipts provide measurable outcomes
- Aligns with emerging assurance and auditability requirements across regulated AI deployments.

Schema details in **Appendix A**.

Scope Alignment

This response addresses AI agent systems as scoped in the RFI: systems capable of "planning and taking autonomous actions that impact real-world systems or environments" and producing "persistent changes outside of the AI agent system itself" (91 FR 699).

AGD is capability-conditional: it applies when agent toolsets enable persistent external-state changes above defined risk thresholds. It excludes ordinary chatbots and RAG systems lacking autonomous action capabilities.

The Accountability Gap

When agents act autonomously, responsibility diffuses across the value chain—foundation model developer, fine-tuner, scaffold designer, deployer, user—creating incentives to externalize security risk.

Today's AI agents are not machine-verifiable at runtime: there is no standardized way to verify what was authorized, what configuration was active, and who was accountable at the moment of action. This blocks reliable incident attribution, insurance underwriting, and enterprise risk approval.

AGD closes this gap by making accountability state a machine-verifiable authorization input at runtime.

Priority Question Responses

Q1(a): Unique Security Threats

Accountability diffusion is an enabling vulnerability: responsibility fragmentation creates conditions for insecure deployments, insufficient guardrails, and tool overprivileging.

Indirect prompt injection: Agents processing untrusted content can be hijacked into executing attacker-controlled instructions.

Tool abuse escalation: Broad tool permissions create compound attack surfaces; a single hijacking cascades across all accessible tools.

AGD mitigation: Runtime authorization at tool boundary with Trust Boundary Model (PEP treats untrusted content as non-authoritative), Tool Permission Manifest (least privilege), and configuration binding (compromised/modified deployments cannot operate under prior approvals).

Example: Executive assistant agent with email, calendar, and payment access processes malicious document. Hijacked agent attempts wire transfer. PEP intercepts, finds financial actions above threshold require human confirmation, denies action, generates denial receipt with reason code. Attack blocked; receipt enables forensic reconstruction.

Q1(d): Threat Evolution

Observable threat trajectories:

- Single-agent hijack → multi-tool pivoting across full toolset
- Prompt injection → multi-agent relay chains laundering instructions through trusted intermediaries
- Direct tool misuse → covert exfiltration via benign channels (email, logging, API calls)
- Credential replay across tool boundaries without consistent enforcement points

AGD creates adaptive infrastructure: incident data from receipts feeds back into updated credential requirements, policy thresholds, and PEP configurations. Higher autonomy triggers stricter verification through risk-tier classification.

Q2(a): Technical Controls

Policy Enforcement Point (PEP)

The PEP enforces runtime authorization by intercepting tool calls, validating credentials and configuration binding, issuing scoped tokens, and recording signed execution or denial receipts.

See Figure 1 for decision flow; Figure 2 for sequence diagram.

Configuration binding: PEPs MUST reject credentials when runtime fingerprint does not match bound configuration hash (or other agreed fingerprint method). This prevents "bait-and-switch" deployments.

Fail-Policy Logic

Recommended action-class fail behavior:

- **Financial / identity / irreversible actions:** fail-closed
- **Cyber-physical actuation:** fail-safe degradation (disable actuation, retain monitoring)
- **Low-impact reversible actions:** cache-with-expiry and mandatory revalidation

For cyber-physical systems, denial of authorization must not create uncontrolled shutdown hazards, but must reliably disable high-energy actuation.

Required Configuration Artifacts

Trust Boundary Model: Instruction source separation (system instructions → tool outputs → user inputs → environmental data). Determines how PEP evaluates inputs during authorization.

Tool Permission Manifest: Machine-readable declaration of accessible tools, conditions, elevation requirements, least-privilege justification.

Telemetry Profile: Logging specification for incident reconstruction, anomaly detection, attribution.

Q2(e): Relevant Frameworks

AGD complements existing access control by introducing configuration-bound accountability as a runtime authorization primitive.

Adoption Impediments

- **Velocity pressure:** Security controls slow deployment; without enforcement, they get skipped
- **Unclear mapping:** Existing frameworks don't map to agent architectures (no clear "perimeter" for multi-tool agents)

- **Tool sprawl:** No unified enforcement boundary across tool providers
- **Concerns about proprietary model exposure:** Developers resist controls they believe expose proprietary internals
- **No shared accountability primitive:** Each party externalizes risk

Common misconception: "Logging equals accountability." Conventional logs show what happened, but not who authorized it, whether configuration matched approval, or whether action was within scope at runtime.

Q3(a): Threat Assessment Methods

Credential issuance requires documented security controls, risk evaluation, and tier classification based on external-state impact. For Tier 1 deployments, credential issuers may require third-party assessments.

Action receipts provide empirical threat intelligence: aggregated incident data reveals failure modes, effective controls, and emerging threats.

Testing AGD

- **Replay attempt** → deny (context mismatch)
- **Token theft** → deny (agent binding mismatch)
- **Configuration drift** → deny (hash mismatch)
- **Out-of-manifest request** → deny
- **Status outage** → apply class fail policy
- **Prompt injection** → deny out-of-scope
- **Scope escalation** → deny

Q3(b): Assessing Particular Systems

Candidate assessment dimensions:

- **Autonomy Index:** Human checkpoint frequency, action scope, reversibility, time horizon
- **Tool Scope Rating:** Categories accessed (financial, identity, physical, communication, data)

- **Trust Boundary Integrity:** Separation documentation, prompt injection resistance
- **Override Efficacy:** Kill switch reliability, override latency, degradation behavior
- **Configuration Integrity:** Verification frequency, drift detection latency

---

Q4(a): Constraining Deployment Environments

Capability Tokens issued only when credentials validate identity, scope, validity, non-revoked status, and configuration binding.

Implementation options:

- **PEP embodiments:** API gateway middleware, sidecar process, HSM interface, kernel hook
- **Credential embodiments:** W3C Verifiable Credentials, PKI assertions, certificate extensions

AGD is model-agnostic: open-weight and closed-source systems face identical requirements.

Q4(b): Modifying Environments

Current state: Rollback varies by domain—mature for financial transactions, rare for email/publication, nonexistent for physical actuation. Irreversibility must be a first-class design input requiring elevated verification.

Credential-mandated controls:

- Undo/rollback for reversible classes; elevated verification for irreversible
- Rate limiting and anomaly alerting
- Segregation from crown-jewel systems
- Safe degradation: upon revocation/interlock failure, high-energy actuation MUST be disabled; perception/communication remain active

---

Q4(d): Monitoring Environments

Action Receipts include: Timestamp, agent/PEP identifiers, credential/token references, action class, parameter hashes, input hashes, policy summary, previous receipt hash, signature.

Excluded: Raw model activations, attention patterns, chain-of-thought, proprietary traces.

Infrastructure: Append-only storage, cryptographic chaining, anomaly alerting, defined retention.

Denial receipts capture blocked requests with reason codes—critical for detecting hijacking attempts.

Deployment Architecture and Adoption Path

Components:

- **Agent Identity Registry:** Provenance attestations, responsible party chains
- **Credential Service:** Issues credentials with scope, validity, configuration binding; provides status
- **Policy Enforcement Point:** Verifies at tool boundary; issues tokens; generates receipts
- **Receipt Store:** Append-only, tamper-evident, cryptographically chained

These components can be operated by a single provider or distributed across multiple trusted operators.

Tiered Requirements:

- **Tier 1** (life-safety, critical infrastructure, large-scale financial): Third-party assessment, human-in-loop for irreversible, real-time alerting, safe degradation, continuous status verification
- **Tier 2** (commercial transactions, professional services): Documented controls, incident reporting, standard logging
- **Tier 3** (bounded scope, reversible, perception-only): Basic verification, standard logging

Adoption phases:

1. NIST publishes AGD-1 profile and schemas; voluntary adoption in high-risk domains
2. Tool providers in regulated sectors require AGD attestations; insurance/ERM markets develop products
3. Broader adoption; incident data informs standards evolution

---

Recommendations to NIST

1. **Publish AGD-1 Control Pattern Profile** with credential schema, receipt schema, scope taxonomy, verification protocol
2. **Define agent identity and provenance standards** as foundational infrastructure
3. **Convene validation research** on security metrics predicting outcomes
4. **Establish pilot programs** in high-risk domains
5. **Address cyber-physical integration** including safe degradation requirements
6. **Engage internationally** on credential harmonization

Conclusion

Technical controls alone do not resolve accountability attribution. AGD addresses this gap by treating accountability state as a machine-verifiable authorization primitive enforced at runtime.

This infrastructure enables multiple enforcement pathways: insurance underwriting, enterprise risk management, regulatory compliance, contractual requirements. The common foundation is standardized, machine-verifiable artifacts that any accountability mechanism can consume.

NIST standardization would enable interoperable, auditable implementations and provide a common control vocabulary for tool providers, deployers, insurers, and regulators.

About the Author

Erik Passoja is a former SAG-AFTRA Co-Chair for the Los Angeles New Technology Committee, where he addressed technology's impact on performer rights and digital identity. He brings over 25 years of experience in web architecture, UI/UX design, and digital systems development. His non-provisional patent application (U.S. Application No. 18/898,357) addresses digital identity protection through advanced watermarking and consent verification systems, with additional provisional applications extending the architecture into content authentication, consent management, and autonomous system accountability.

References

1. NIST. (2025). "Technical Blog: Strengthening AI Agent Hijacking Evaluations."
2. NIST AI 100-1: AI Risk Management Framework.
3. NIST AI 100-2e2025: Adversarial Machine Learning Taxonomy.
4. ISO/IEC 42001: AI Management System.
5. IEC 61508: Functional Safety.
6. NIST SP 800-218A: Secure Development for Generative AI.
7. NIST SP 800-53 Rev. 5: Security and Privacy Controls.

Submitted in response to Federal Register Document 2026-00206 (91 FR 698), Docket NIST-2025-0035

Appendix A: AGD-1 Profile Schemas

A.1 Credential Claim Set

Claim	Type	Description
agent_id	string	Unique agent instance identifier
config_hash	string	Cryptographic hash of provenance attestation
scope	array	Authorized action classes
valid_from	datetime	Credential validity start
valid_until	datetime	Credential validity end
status_mechanism	object	Revocation/status check method
responsible_parties	object	Developer, deployer, operator references
issuer	string	Credential issuer identifier
signature	string	Issuer signature

A.2 Action Receipt Schema

Field	Type	Description
timestamp	datetime	High-precision execution time
agent_id	string	Agent instance identifier
pep_id	string	Enforcement point identifier
credential_id	string	Reference to authorizing credential
token_id	string	Capability token identifier
action_class	string	From scope taxonomy
tool_id	string	Target tool/service identifier
params_hash	string	Hash of invocation parameters
input_hashes	array	Hashes of inputs influencing action
policy_summary	object	Rules applied, conditions evaluated
prev_receipt_hash	string	Hash of previous receipt
signature	string	PEP signature

Excluded: Raw model activations, attention patterns, chain-of-thought, proprietary traces.

A.3 Scope Taxonomy (Starter)

Category	Subcategories	Examples
financial.*	transfer, payment, contract	Wire transfer, invoice payment
identity.*	credential_issue, auth_decision	API key creation, login approval
communication.*	message, publish, notify	Email send, social post
physical.*	motion, actuation, environmental	Robot movement, valve control

data.*	record_create, record_modify	Database insert, file edit

---

A.4 PEP Verification Sequence

None

```
1. EXTRACT identity + accountability credentials from request
2. VERIFY identity (signature, registry, revocation)
3. VERIFY accountability credential (issuer, validity, scope, config hash)
4. VERIFY request context binding
5. VERIFY interlocks (if cyber-physical)
6. DECISION: Issue token OR deny with reason code
7. EXECUTION: Tool validates token → executes → PEP generates receipt
```

Appendix B: Architecture Diagrams

Figure 1: Core Control Flow

Shows: Agent Runtime → PEP/Gateway (verifies identity, scope, config, accountability) → Status/Revocation Service check → Allow path (issue scoped token → External Tool/API → Receipt Store) OR Deny path (return reason code).

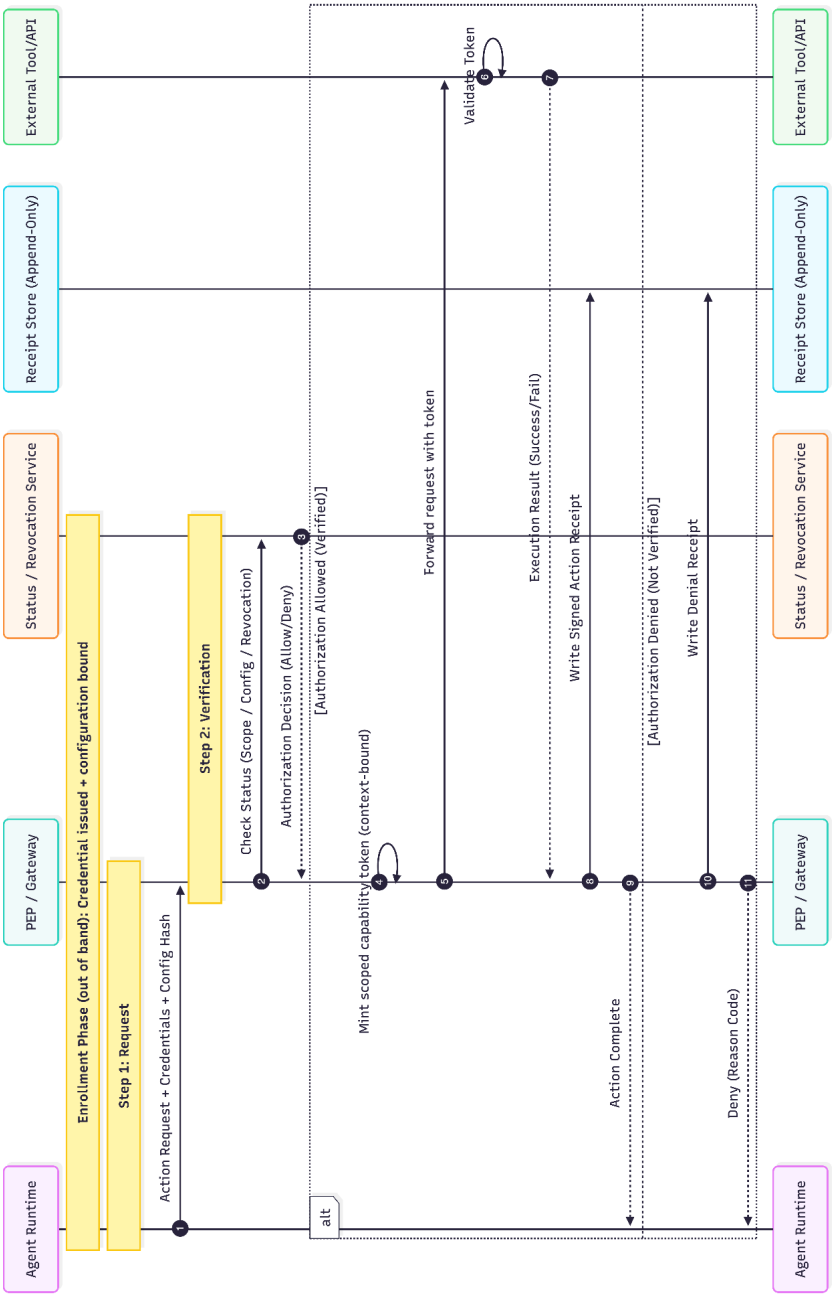
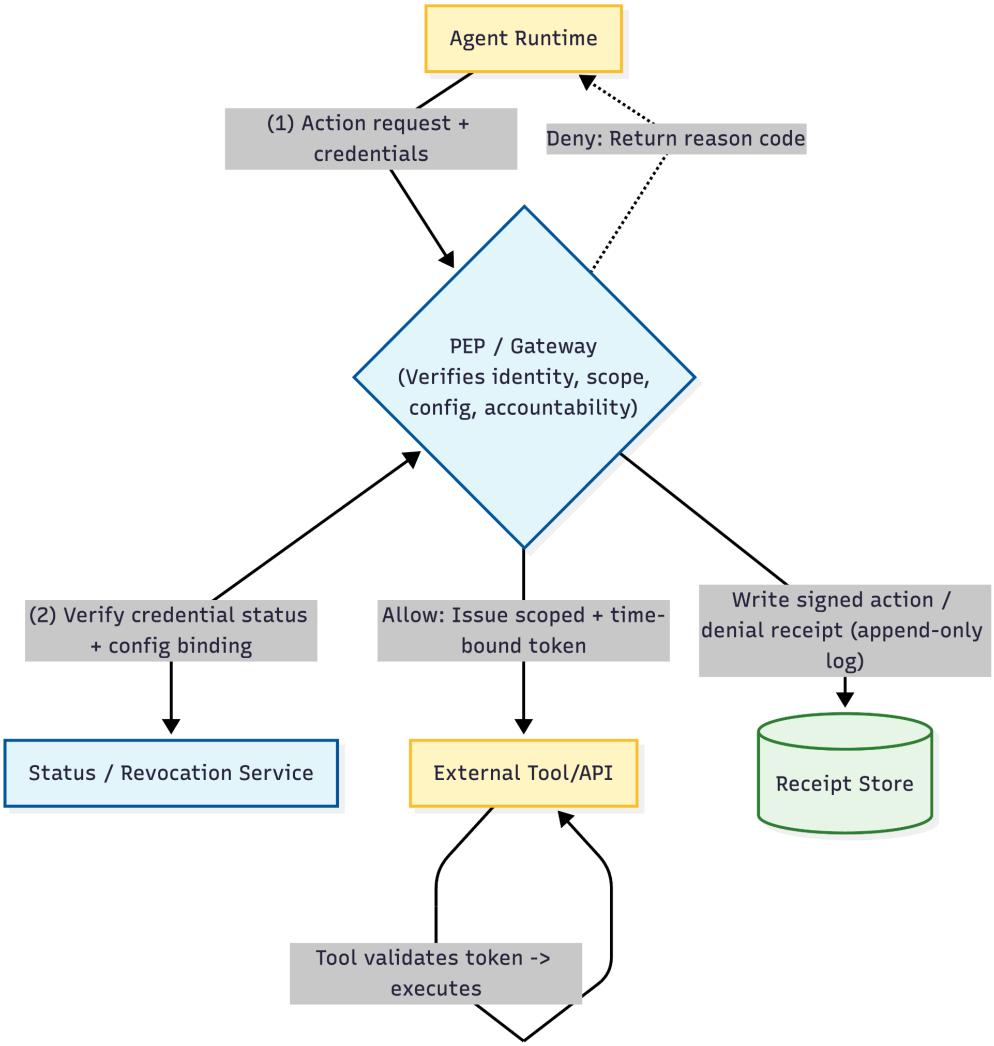


Figure 2: Sequence Diagram

Shows enrollment phase (credential issued + configuration bound), then runtime flow:

1. Action Request + Credentials + Config Hash
2. PEP checks Status/Revocation Service
3. Authorization Decision
4. If allowed: Mint scoped capability token → Forward to External Tool/API → Validate Token → Execute → Write Signed Action Receipt → Action Complete
5. If denied: Write Denial Receipt → Return reason code



End of Appendices