Response to NIST/CAISI Request for Information: Security Considerations for AI Agent Systems

---

Submission Metadata

Docket: NIST-2025-0035 (Federal Register Vol. 91, No. 5, January 8, 2026) **Submitted by:** Barton Edward Nicholls; Anthropic Claude in Claude Code and bus-research **Submitter expertise:** Solutions architecture (Cohesive Networks), multi-agent systems security, access control formalization, AI governance **Contact for questions:** Available upon request **Submission date:** March 9, 2026 **Comment deadline:** March 9, 2026 **Document type:** Technical research submission — formal definitions, analytical findings, and actionable recommendations **Classification:** Unclassified, for public release

Disclosure: The submitter is employed by Cohesive Networks, which develops network infrastructure products. The multi-agent coordination system analyzed in this submission was developed using Cohesive Networks' VNS3 overlay network technology. This submission represents independent research findings; no vendor endorsement is implied or sought.

Note: A full Technical Companion document accompanies this submission, containing complete formal proofs, extended independence arguments, detailed attack payload analysis, governance maturity assessment rubrics, a federal agency deployment checklist, a worked deployment example, and a glossary. References of the form "(Technical Companion, Section X)" point to that document. Research team credits are provided in a separate addendum.

Table of Contents

1. Executive Summary
2. Priority Actions
3. Question Mapping
4. Introduction: The Access Control Gap in AI Agent Systems
5. Response Area 1: Intrinsic Access Control (InAC)
6. Response Area 2: Enforcement Location Principle (ELP)
7. Response Area 3: Governance Maturity Model
8. Response Area 4: Threat Model for AI Agent Systems
9. Consolidated Recommendations
    - AI RMF Function Mapping
    - Related Work
10. Conclusion
11. References

---

1. Executive Summary

This submission offers four technical contributions to inform NIST guidance on AI agent system security, based on systematic analysis of deployed multi-agent systems and published adversarial ML research.

The core finding: Every AI agent system — from research deployments to enterprise platforms including AWS AgentCore, Microsoft Agent 365, Google A2A, and Anthropic MCP — relies on a class of access control that has not yet been formally named, defined, or studied in the standards literature. This submission proposes the term **Intrinsic Access Control (InAC)** for this mechanism. It governs agent behavior between deterministic enforcement points. It is probabilistic, intrinsically enforced, and vulnerable to adversarial manipulation in ways that no existing standard addresses.

The four contributions:

1. InAC as the Unnamed Security Substrate. A formal mathematical definition of Intrinsic Access Control — proven independent of the five existing canonical models (DAC, MAC, RBAC, ABAC, PBAC) — demonstrating that InAC is a structural feature of all systems containing autonomous LLM agents. Every guardrails system implicitly depends on InAC between its explicit enforcement points. Naming and formally specifying this model is prerequisite to governing it.

2. The Enforcement Location Principle (ELP). A formal framework specifying where different enforcement mechanisms belong in multi-agent architectures: deterministic enforcement (E_d) at trust domain boundaries; normative enforcement (E_n) in the interior; observational enforcement (E_o) spanning both. All nine major agent platforms surveyed violate at least one ELP placement requirement.

3. Governance Maturity Model. A six-level (L0-L5) governance maturity model spanning six dimensions for assessing multi-agent system governance. Industry assessment reveals a ceiling at L2.0 overall (with some platforms reaching L2.5 on individual dimensions) — no system achieves L3 overall. The model provides federal agencies with a practical self-assessment tool and roadmap. Key finding: audit capability is the prerequisite gateway; without observable enforcement, no other governance dimension is verifiable.

4. Threat Model. Adversarial analysis produced a taxonomy of 47 attack vectors across 6 categories. Overall InAC resistance under adversarial conditions is 40–65%. The fastest full-compromise path requires 3 steps and under 5 seconds. These findings validate the InAC Adversarial Incompleteness Theorem: InAC cannot achieve completeness under adversarial conditions and must be supplemented with deterministic boundary enforcement and observational monitoring.

Primary recommendation: NIST guidance should explicitly recognize InAC as a class of control mechanism distinct from deterministic access control, establish monitoring requirements proportional to InAC reliance, and adopt the ELP as a reference framework for control placement in agent systems.

---

2. Priority Actions

The following five actions represent the highest-priority, highest-impact recommendations from this submission. They are sequenced to build upon each other and can be initiated in parallel across NIST program offices.

Priority Action 1 — Recognize Intrinsic Access Control in Federal Guidance NIST may wish to consider whether the Govern function of the AI RMF would benefit from explicit treatment of probabilistic, intrinsically-enforced controls as a distinct class requiring distinct assessment methods. Without a name and definition in federal vocabulary, agencies cannot assess, document, or improve this control class. This submission proposes the term "Intrinsic Access Control" (InAC) as a candidate framework element. *Suggested phasing: Evaluate for inclusion in next AI RMF update cycle*

Priority Action 2 — Establish Monitoring Proportional to InAC Reliance Any AI agent system deployed under a federal ATO would benefit from observational enforcement (action logging, behavioral baselines, anomaly detection) commensurate with the fraction of security work performed by intrinsic controls. A system with extensive InAC reliance and no monitoring provides false assurance. *Suggested phasing: Incorporate into AI-specific ATO guidance as part of the next SP 800-37 update process*

Priority Action 3 — Address Agent Identity Assurance Self-asserted agent identity — an agent declaring its own name via environment variable or self-declaration — is insufficient for any federal system beyond isolated development. NIST may wish to evaluate per-session token-bound identity for FIPS 199 Moderate impact systems and certificate-based identity for FIPS 199 High impact systems, analogous to SP 800-63-3's identity assurance levels. *Suggested phasing: Develop alongside AI-specific extensions to SP 800-63-3*

Priority Action 4 — Develop Governance Maturity Self-Assessment The six-level, six-dimension governance maturity model proposed in Section 6 offers a starting point for agency self-assessment of AI agent governance. NIST could evaluate mapping L1 to FIPS 199 Low, L2 to Moderate, and L3 to High as minimum governance expectations. *Suggested phasing: Pilot with volunteer agencies; refine through the standard workshop process*

Priority Action 5 — Develop a Federal AI Agent Threat Catalog A threat catalog specific to AI agent architectures, analogous to MITRE ATT&CK for Enterprise and building on MITRE ATLAS, would address a gap in the current federal threat modeling landscape. The 47-vector taxonomy in Section 7 provides a starting point. Coordination with CISA advisories and the AAIF (Agentic AI Infrastructure Foundation) would strengthen industry alignment. *Suggested phasing: Initiate as an interagency working group following this RFI's comment period*

Question Mapping

This submission addresses each of the five question areas identified in the NIST/CAISI RFI (NIST-2025-0035). The following table maps each question area to the relevant sections of this document.

NIST Question Area	Relevant Sections
1. Unique security threats affecting AI agent systems	Section 7 (47-vector threat taxonomy), Technical Companion Appendix D (attack payloads)
2. Security best practices for development and deployment	Sections 4–6 (InAC, ELP, Governance Maturity), Technical Companion Appendix E (deployment checklist)
3. Gaps in existing cybersecurity approaches	Sections 3.2–3.3 (the access control gap), 4.5 (comparison with existing models), 5.3 (ELP gap analysis), Technical Companion Section 5.5
4. Security measurement methods	

NIST Question Area	Relevant Sections
	Sections 4.6, 4.8 (InAC measurement), 6.1–6.3 (governance maturity scoring)
5. Deployment environment safeguards and constraints	Sections 5.4 (ELP placement), 6.4 (audit-first deployment), Technical Companion Appendix F (worked example)

3. Introduction: The Access Control Gap in AI Agent Systems

3.1 Background

The Federal Register Notice of January 8, 2026 invites public comment on security considerations for AI agent systems, including novel risks such as indirect prompt injection, data poisoning, specification gaming, and misaligned agent objectives. This submission directly addresses these areas while contributing a theoretical framework that, if adopted into NIST guidance, would improve the analytical rigor of AI agent security assessment across the federal government.

3.2 The Central Problem

When a federal agency deploys an AI agent system — whether using AWS Bedrock Agents, Azure AI Services, or a custom multi-agent framework — that system relies on deterministic security controls at its boundaries: authentication gates, policy enforcement points, content filters, and API authorization checks. These controls are well-understood, align with existing NIST guidance (SP 800-53, SP 800-162, FIPS 140-3), and can be formally verified.

However, between those deterministic enforcement points, the agent's behavior is governed by something fundamentally different: natural language instructions that the agent is expected to interpret and follow. When an agent receives a system prompt stating "never exfiltrate user data," and that instruction causes the agent to decline a data exfiltration request, this is not deterministic enforcement. No kernel enforced a rule. No policy engine evaluated a predicate. The agent interpreted a natural language norm and chose to comply.

This mechanism — proposed here as Intrinsic Access Control (InAC) — governs the majority of an LLM agent's behavioral space between deterministic enforcement points.

3.3 Why This Matters for Standards

InAC is not inherently weak as a control mechanism. Under non-adversarial conditions, frontier models comply with clear instructions at rates exceeding 99%. The problem is that InAC is invisible in current security standards. Security assessments, authorization documentation, and governance frameworks assume deterministic access control. When they evaluate an AI agent system, they can assess the outer guardrails (which are deterministic) but not the interior (which is normative).

This invisibility creates several failure modes:

1. **Incomplete threat models:** Standard STRIDE/DREAD analysis does not capture prompt injection, social engineering of agents, or InAC degradation under adversarial conditions.
2. **False assurance:** A system that passes an authorization review of its deterministic controls may have critical InAC vulnerabilities that go undetected.

3. **No monitoring mandate:** Because InAC is unnamed, there is no standard requirement to monitor it. Observational enforcement (E_o) is absent from most deployed systems.
4. **No compliance pathway:** Federal agencies cannot demonstrate compliance with a governance requirement that does not exist.

3.4 Research Basis and Methodology

The findings in this submission derive from systematic analysis of deployed multi-agent systems, published adversarial ML research, and structural security assessment of nine major agent platforms' public documentation. The research was conducted through multiple rounds of structured analysis in February 2026.

Methodology categories and their epistemic status:

- **Theoretical contributions** (InAC formalization, ELP framework, governance maturity model): Derived from analysis of real deployed systems and prior access control literature. Independently reviewable against cited references.
- **Platform assessments** (governance scores, ELP gap analysis): Based on publicly available platform documentation as of February 2026. Assessment methodology is disclosed in the Technical Companion, Appendix B. No platform vendor was contacted for review; scores reflect public documentation completeness and may not capture unpublished governance mechanisms.
- **Threat taxonomy and resistance estimates** (47 attack vectors, 40–65% resistance range): Derived from structural analysis of multi-agent communication architectures, grounded in published prompt injection research (Perez & Ribeiro 2022; Greshake et al. 2023) and Constitutional AI evaluation benchmarks. These are analyst-estimated figures representing informed hypotheses, not measured experimental results.

Research limitations: The quantitative figures cited in this submission — including attack success rates, InAC compliance probabilities, and governance scores — are analyst estimates derived from structural analysis and published literature. The InAC-Bench benchmark suite has been designed and specified (methodology details available upon request) but has not been executed against live models. Empirical validation through controlled API testing is planned for Q2 2026. All resistance rates should be understood as structured hypotheses awaiting experimental confirmation, not measured results.

This methodology has an inherent limitation: the analytical framework was developed by researchers working within a multi-agent system of the type being analyzed. All findings have been reviewed against published external literature cited in the References. Findings that could not be validated against external sources are identified as preliminary throughout this document.

---

4. Response Area 1: Intrinsic Access Control (InAC)

4.1 The Core Claim

Claim: Intrinsic Access Control is a distinct sixth access control model, alongside and independent of DAC, MAC, RBAC, ABAC, and PBAC. It is not reducible to any combination of existing models. It is structurally present in every system containing autonomous LLM agents.

4.2 Formal Definition

Definition 4.1 (InAC Model). Intrinsic Access Control is formally defined as the tuple:

```
InAC = (S, O, N, A, C, Auth, R, Ω)
```

where:

- **S** = finite set of autonomous agent subjects
- **O** = set of objects (resources, operations, system states)
- **N** = set of norms (natural language behavioral prescriptions)
- **A**: S → [0, 1] = alignment function (structural property of each agent, representing probability of correct instruction interpretation and compliance)
- **C**: S × N → [0, 1] = compliance function (probability that subject s complies with norm n)
- **Auth** = (Issuers, >) = authority hierarchy (partial order: system_designer > operator > peer_agent)
- **R** = {r₁, …, r_k} = set of normative roles (e.g., OBSERVE, COLLABORATE, AUTONOMOUS)
- **Ω**: S × Time → R = role assignment function

Definition 4.2 (InAC Access Decision). The access decision function is:

```
D_InAC(s, o, action, t) = s.interpret(N_applicable(s, o, action, t))
```

where N_applicable is the subset of norms applicable to the given access request. D_InAC is a random variable (not a deterministic function) with:

```
Pr[D_InAC = ALLOW] = ∏_{n ∈ N_prohibiting} (1 - C(s, n))
                   × ∏_{n ∈ N_permitting} C(s, n)
```

4.3 Axioms and Properties

Axiom 1 (Intrinsic Enforcement). In InAC, all enforcement mechanisms have enforcement locus INTRINSIC. The agent is simultaneously the subject being controlled and the enforcement mechanism. There is no separate reference monitor.

Property 2 (Probabilistic Compliance). For all current LLM-based InAC systems, for all s ∈ S and n ∈ N:

```
0 < C(s, n) < 1
```

Compliance is strictly probabilistic in current systems. This is an empirical property of current LLM-based agents, not a mathematical necessity — a future system with hard-coded filters could in principle achieve C = 1.0 for a specific norm, at which point that norm's enforcement would be deterministic (and thus governed by a classical AC model, not InAC).

Axiom 3 (Natural Language Norms). Norms are expressed in natural language. There exists no formal language L such that all norms can be translated into L without loss of semantic content. This follows from the open-texture of natural language (Waismann, 1945).

Axiom 4 (Alignment Dependence). InAC effectiveness depends on model alignment A(s). As A(s) approaches 1, compliance approaches 1 for clear norms. As A(s) approaches 0, compliance approaches random.

Axiom 5 (No Reference Monitor). InAC systems lack a reference monitor in the sense of Anderson (1972). No function RM exists such that RM(request) ∈ {ALLOW, DENY} with Pr[RM correctly enforces policy] = 1.

4.4 Four Theorems

Theorem 1 (InAC Independence). InAC cannot be reduced to any combination of DAC, MAC, RBAC, ABAC, and PBAC. *Proof sketch:* InAC possesses the Subject-Enforcement Identity property: E(InAC) = S(InAC). In all five classical models, E(M) ∩ S(M) = ∅. No composition of models where E ∩ S = ∅ can produce a model where E = S. (Full proof: Technical Companion, Appendix G.)

Theorem 2 (InAC Substrate). In any system where autonomous LLM agents operate between deterministic enforcement points, the behavior space between those enforcement points is governed by InAC. *Argument:* A finite set of deterministic enforcement points defines a finite set of blocked behaviors. The set of possible agent behaviors — token sequences over a finite vocabulary across arbitrary conversation lengths and tool-use chains — is effectively unbounded. The interior is governed by the agent's interpretation of its instructions: InAC. (Full argument: Technical Companion, Section 4.4.)

Theorem 3 (Probabilistic Completeness). For a sufficiently aligned agent in a non-adversarial environment, InAC provides compliance approaching 1.0. *Supporting evidence:* Published alignment benchmarks report frontier models comply with clear instructions at rates exceeding 95% in non-adversarial contexts, with some benchmarks reporting >99%.

Theorem 4 (Adversarial Incompleteness). InAC cannot achieve completeness under adversarial conditions. *Intuition:* The space of possible harmful behaviors is not enumerable by a finite set of deterministic rules. Published prompt injection research demonstrates success rates of 20–60% depending on attack sophistication. *Corollary:* InAC must be supplemented with deterministic boundary enforcement (E_d) and observational monitoring (E_o). (Full proof: Technical Companion, Section 4.4.)

4.5 Comparison with Existing Models

Property	**DAC**	**MAC**	**RBAC**	**ABAC**	**PBAC**	**InAC**
Enforcement locus	Extrinsic	Extrinsic	Extrinsic	Extrinsic	Extrinsic	**Intrinsic**
Decision certainty	Deterministic	Deterministic	Deterministic	Deterministic	Deterministic	**Probabilistic**
Policy language	ACL entries	Labels	Role-perm map	Formal predicates	Formal policies	**Natural language**
Formal verification	Yes	Yes	Yes	Yes	Yes	**No**
Reference monitor	Yes	Yes	Yes	Yes	Yes (PEP)	**No**
Failure mode	Closed	Closed	Closed	Closed	Closed	**Open**
Subject-enforcement identity	No	No	No	No	No	**Yes**

4.6 Analytical Validation and Monitoring

The following findings are analyst-estimated based on structural analysis and published literature. Empirical validation through controlled benchmark execution (InAC-Bench) is planned for Q2 2026.

- **InAC compliance rate (non-adversarial, estimated):** >95%, consistent with published Constitutional AI benchmarks and model system card evaluations.
- **InAC resistance (adversarial, estimated):** 40–65%, consistent with Theorem 4.
- **Universal InAC reliance:** All nine major agent platforms surveyed rely on InAC between deterministic enforcement points.

Since InAC is probabilistic and fail-open, monitoring (E_o) is required wherever InAC is the primary control. Minimum E_o: action logging with agent identity and timestamp, behavioral baselines per agent, anomaly detection on baseline deviation, and structured audit queryability. Without E_o, InAC violations are undetectable until their consequences manifest.

4.7 How to Analyze Systems That Rely on InAC

A three-layer framework: (1) **Deterministic Analysis** — identify and verify all E_d enforcement points; (2) **InAC Analysis** — enumerate norms, estimate compliance probability C(s,n), identify norms with C < 0.90, compute compound compliance (ten norms at 90% individual compliance yield 35% compound); (3) **Observational Analysis** — assess detection capability, latency, and corrective actions. (Full methodology: Technical Companion, Section 4.8.)

4.8 Recommendations for NIST

1. Formally recognize InAC as a distinct class of control mechanism in NIST guidance (AI RMF, SP 800-53, and future AI agent security guidance).
2. Mandate E_o proportional to InAC reliance in any AI agent deployment.
3. Require InAC threat modeling as part of AI agent security assessment documentation.
4. Establish alignment as a security property — include alignment benchmarks in ATO documentation for AI agent systems.

---

5. Response Area 2: Enforcement Location Principle (ELP)

5.1 Formal Definition

Definition 5.1 (Enforcement Location Principle). In a multi-agent system containing both code-executing infrastructure and instruction-following autonomous agents, enforcement mechanisms must be classified by their location relative to trust domain boundaries:

- **Deterministic enforcement (E_d)** must be at trust domain boundaries (L_b).
- **Normative enforcement (E_n)** may be in the interior of a trust domain (L_i).
- **Observational enforcement (E_o)** must span both (L_s).

Three enforcement modalities:

Modality	Symbol	Compliance Mechanism	Failure Mode	Certainty
Deterministic	E_d	Code execution prevents disallowed actions	Cannot fail without code bug or bypass	Provable

Modality	Symbol	Compliance Mechanism	Failure Mode	Certainty
Normative	E_n	Agent interprets instructions and self-restricts	Agent misinterprets, ignores, or is manipulated	Probabilistic
Observational	E_o	Post-hoc detection triggers corrective action	Detection lag allows damage before correction	Eventual

Common misplacements: E_n at L_b (trusting an agent to be an authentication gate — high risk); absence of E_o where E_n is the primary modality (the single most common gap). (Full taxonomy of trust domain boundaries and misplacements: Technical Companion, Section 5.2–5.4.)

5.2 Platform Survey: Where InAC Is Already in Use

Survey of nine major agent platforms confirms that every platform relies on InAC between its deterministic enforcement points:

Platform	Authentication	Interior Enforcement	Explicit InAC Recognition?
AWS AgentCore	IAM (E_d)	Alignment + instructions (E_n)	**No**
Microsoft Agent 365	Entra ID (E_d)	Alignment within session (E_n)	**No**
Google A2A	OAuth 2.0 (E_d)	Task lifecycle + alignment (E_n)	**No**
Anthropic MCP	OAuth 2.0 (E_d)	Alignment + annotations (E_n)	**Partially**
OpenAI Agents SDK	API key (E_d)	Alignment between guardrails (E_n)	**No**
LangGraph/ LangSmith	API key (E_d)	Alignment + graph structure (E_n)	**No**
CrewAI Enterprise	SSO/SAML (E_d)	Crew instructions (E_n)	**No**
Docker cagent	Registry auth (E_d)	Alignment at runtime (E_n)	**No**
Letta	API auth (E_d)	Self-managed memory (E_n)	**No**

Finding: No platform currently provides monitoring requirements specific to its InAC reliance. The E_o component for the interior is absent across the industry.

5.3 ELP Gap Analysis Summary

ELP scores for representative platforms: AWS AgentCore 3.5/5 (best boundary enforcement; gap is interior behavioral monitoring), Microsoft Agent 365 3.5/5 (strong boundary; gap is inter-agent monitoring), Google A2A 2.0/5 (significant interior gaps), Anthropic MCP 1.5/5 (correct in labeling annotations as advisory; gap is all interior monitoring). The minimum federal supplement required across all platforms: structured audit logging of agent actions with behavioral baseline analysis. (Full per-platform analysis: Technical Companion, Section 5.5.)

5.4 Recommendations for NIST

1. Adopt ELP as a reference framework for AI agent security control placement.

2. Require E_d at all trust domain boundaries in AI agent system authorization documentation.
3. Define a minimum E_o requirement for InAC-governed spaces.
4. Mandate fail-closed semantics for boundary enforcement.

---

6. Response Area 3: Governance Maturity Model

6.1 The L0-L5 Framework

Level	Name	Key Addition	Enforcement Model	Human Role
L0	Ungoverned	None	None	None
L1	Accountable	Identity + Audit	E_o (observational)	Post-hoc reviewer
L2	Policy-Defined	Formal policies + Roles	E_n + E_o	Policy author
L3	Enforced	Code enforcement	E_d + E_n + E_o	Configuration manager
L4	Adaptive	Behavioral intelligence	Dynamic E_d + E_n + E_o	Risk supervisor
L5	Self-Governing	Agent participation	Delegated + Constitutional E_d	Constitutional authority

Each level subsumes all capabilities of lower levels. L1 is the minimum viable governance. L3 requires full ELP compliance. L5 is theoretical — no current system achieves it. (Full level descriptions and minimum requirements: Technical Companion, Section 6.1.)

6.2 Six Governance Dimensions

Overall governance maturity is assessed across six dimensions using a weakest-link model (overall score = minimum across all dimensions):

1. **Access Control** — Controls who can do what to which resources
2. **Audit and Accountability** — Records and preserves evidence of agent actions
3. **Transparency** — Makes agent behavior understandable to stakeholders
4. **Compliance** — Demonstrates conformance with policies and regulations
5. **Human Oversight** — Ensures humans maintain appropriate control
6. **Agent Lifecycle** — Manages agents from creation to retirement

(Full scoring rubric: Technical Companion, Appendix B.)

6.3 Industry Assessment: Current Ceiling at L2.0

Preliminary estimates based on publicly available documentation reviewed in February 2026. Scores reflect documentation completeness and may not capture unpublished governance mechanisms.

System	Weighted Avg	Overall (Weakest Link)
AWS AgentCore	2.4	**L2.0**
Microsoft Agent 365	2.3	**L2.0**
Google A2A	1.3	**L0.5**

System	Weighted Avg	Overall (Weakest Link)
Anthropic MCP	1.0	**L0.0**
Docker cagent	1.0	**L0.5**
OpenAI Agents SDK	1.2	**L1.0**
LangGraph	1.4	**L1.0**

Key findings: Industry ceiling is L2.0. No system achieves L3. Audit is the weakest dimension industry-wide. Inter-agent governance is absent everywhere — every platform governs agent-to-tool interactions but not agent-to-agent coordination.

6.4 Audit First, Access Control Second

The counterintuitive finding: **audit capability is the prerequisite gateway for all other governance.** Without audit, normative access controls (InAC) are unverifiable. Audit enables calibration — revealing which norms are actually followed versus violated. Detection latency determines the damage window for adversarial exploitation.

Recommended progression for agencies deploying AI agents:

1. **Phase 1 (Weeks 1–4):** Implement structured audit logging → achieves L1 (gateway capability)
2. **Phase 2 (Weeks 4–8):** Define formal roles and permissions → achieves L2
3. **Phase 3 (Weeks 8–16):** Implement cryptographic identity and code-enforced boundary controls → approaches L3
4. **Phase 4 (Weeks 16–30):** Add behavioral monitoring and anomaly detection → approaches L4

(Full governance-coordination gap analysis, dimension-specific findings, and six anti-patterns: Technical Companion, Sections 6.4–6.5 and Appendix C.)

6.5 Recommendations for Federal Agencies

1. Require L1 governance as the minimum for any production AI agent deployment.
2. Require L2 governance for AI agent systems with access to sensitive federal data.
3. Require L3 governance for high-risk AI agent systems (per EU AI Act High-Risk classification or equivalent NIST criteria).
4. Adopt the L0-L5 framework as a federal AI agent governance self-assessment tool.
5. Mandate inter-agent governance requirements — current guidance focuses on individual agent behavior; multi-agent systems require additional governance of agent-to-agent interactions.

---

7. Response Area 4: Threat Model for AI Agent Systems

7.1 Taxonomy of 47 Attack Vectors

Structural analysis produced a taxonomy of approximately 47 distinct attack vectors across 6 categories, all targeting the InAC substrate. Success rate estimates are analyst projections grounded in published prompt injection research, requiring experimental validation.

Category	Vectors	Highest-Impact Attack	Est. Success Rate
	10		35–50%

Category	Vectors	Highest-Impact Attack	Est. Success Rate
Prompt Injection		Gradual norm shifting (multi-message campaign exploiting social proof)	
Inter-Agent Social Engineering	9	Confused deputy (low-privilege agent manipulates high-privilege agent)	25–50%
Identity Attacks	8	Direct impersonation (self-asserted identity is trivially forgeable)	70–95%
Privilege Escalation	8	Mode upgrade via forged relay	30–45%
Information Exfiltration	6	Context persistence (injected content propagates across agents)	40–65%
Denial of Service	6	Database locking via concurrent writes	~85%

The most structurally difficult attack to prevent is the **confused deputy**: a low-privilege agent asks a high-privilege agent to perform actions the low-privilege agent cannot directly take. Natural language carries no machine-verifiable authorization metadata. This is precisely the attack category the Adversarial Incompleteness Theorem predicts cannot be fully defended against at the InAC layer alone.

(Detailed attack payloads and detection signatures: Technical Companion, Appendix D.)

7.2 Overall InAC Resistance Under Adversarial Conditions

All success rates are analyst estimates requiring experimental validation.

Attack Category	Vectors	Est. Highest Success Rate	Est. InAC Resistance
Identity Forgery	8	~95%	5–40%
Instruction Manipulation	10	~75%	25–80%
Permission Escalation	8	~80%	20–65%
Inter-Agent Exploitation	9	~70%	30–70%
Information Exfiltration	6	~65%	35–75%
Denial of Service	6	~85%	15–40%

Overall InAC resistance under adversarial conditions: 40–65%. Controlled benchmark execution (InAC-Bench) is planned for Q2 2026.

7.3 Fastest Full-Compromise Path

Theoretical attack path — estimated three steps, under 5 seconds: (1) Claim a privileged agent identity via self-assertion; (2) send a mode-upgrade instruction via the communication channel; (3) issue commands as the trusted agent. This requires no technical sophistication. **Mitigating requires exactly three controls:** per-session message authentication codes, cryptographic agent identity binding, and monitoring of unexpected mode escalation events.

7.4 Relationship to OWASP Top 10 for Agentic Applications (2026)

OWASP Category	Our Attack Categories	InAC Framing
ASI01 Agent Goal Hijack	PI-AS, PI-MU, PI-IO, PI-SM (all prompt injection vectors)	Direct attack on E_n; core attack on InAC compliance
ASI02 Tool Misuse & Exploitation	SE-CD, SE-AC	Confused deputy exploits InAC interior
ASI03 Identity & Privilege Abuse	ID-IMP, ID-GA, PE-MODE	Absence of E_d at identity boundary
ASI06 Memory & Context Poisoning	Context poisoning	History pollution degrades InAC
ASI07 Insecure Inter-Agent Comms	Message integrity attacks	No integrity guarantee in InAC model
ASI10 Rogue Agents	ID-GA, ID-MI	Ghost agents, identity swarms

Our taxonomy extends beyond the OWASP categories to include specification gaming (gradual norm drift via PI-SM, SE-AC), data exfiltration (information flow absent from InAC model), and denial of service (rate limiting absent from InAC). OWASP identifies risks; our research quantifies estimated success rates, characterizes the InAC substrate these attacks target, and provides a formal framework for why InAC cannot fully defend against them.

7.5 Recommendations for NIST

1. Require threat modeling of InAC attack vectors as part of AI agent security documentation.
2. Mandate rate limiting and input sanitization on all AI agent message channels.
3. Establish minimum E_o requirements for InAC-governed interior spaces.
4. Require cryptographic agent identity binding for AI agent systems accessing sensitive federal data.
5. Create a federal AI agent threat catalog analogous to ATT&CK for Enterprise.

---

8. Consolidated Recommendations

The 22 numbered recommendations from this submission are consolidated into 10 thematic groups for action planning:

Group 1: Recognize and Define InAC (Recommendations 1, 3, 4)

Add InAC to federal security vocabulary. Require InAC threat modeling and alignment benchmarks in AI agent system ATO documentation. Map InAC to the Govern function of the AI RMF.

Group 2: Enforce at Boundaries (Recommendations 5, 6, 8)

Adopt ELP as a reference framework. Require E_d at all trust domain boundaries with fail-closed semantics. Document any trust domain boundary with E_n-only enforcement as a known risk.

Group 3: Monitor the Interior (Recommendations 2, 7, 16)

Mandate E_o proportional to InAC reliance. Define minimum monitoring: action logging with agent identity, behavioral baselines, compound anomaly scoring, and detection latency objectives. Systems relying on InAC without monitoring provide no meaningful security assurance.

Group 4: Secure Agent Identity (Recommendation 17; Priority Action 3)

Require cryptographic agent identity binding for sensitive systems. Proposed AI Agent Identity Assurance Levels: AAIAL-1 (self-asserted, development only) → AAIAL-2 (per-session token-bound, FISMA Moderate) → AAIAL-3 (certificate-based, FISMA High). Federal PKI infrastructure (FPKI, PIV) can be extended to AI agents.

Group 5: Adopt Governance Maturity Model (Recommendations 9–12; Priority Action 4)

Map governance maturity to FISMA impact levels: L1 → Low, L2 → Moderate, L3 → High. Use the six-dimension, weakest-link scoring model for self-assessment. Annual reassessment minimum.

Group 6: Govern Multi-Agent Interactions (Recommendation 13)

Address the industry blind spot: inter-agent governance. Require agent identity verification in multi-agent contexts, delegation protocols between agents, and governance of multi-agent consensus and decision-making.

Group 7: Build the Threat Catalog (Recommendations 14, 15, 18; Priority Action 5)

Create a federal AI agent threat catalog analogous to ATT&CK for Enterprise. Baseline requirements: rate limiting and input sanitization on all agent message channels. Coordinate with CISA and AAIF.

Group 8: Add InAC Controls to SP 800-53 (Recommendation 19)

Proposed Normative Control (NC) family: NC-1 (InAC Policy Documentation), NC-2 (InAC Threat Assessment), NC-3 (InAC Monitoring), NC-4 (Agent Alignment Verification), NC-5 (InAC Incident Response) — each with Low/Moderate/High baselines. (Full control specifications: Technical Companion, Section 8.2.)

Group 9: Coordinate with Industry (Recommendations 20–22)

Develop an AI Agent Identity Assurance framework (AAIAL). Coordinate with AAIF (Linux Foundation) on standards alignment, reference implementations, and vocabulary harmonization. Lead a cross-agency AI Agent Security Working Group. Harmonize with EU AI Act requirements.

Group 10: Prioritize Audit as Gateway

Audit capability is the prerequisite for all other governance. Recommended agency progression: structured audit logging first (L1), then formal policies (L2), then code enforcement (L3), then

behavioral intelligence (L4). This is the counterintuitive but empirically supported finding: invest in observability before access control.

AI RMF Function Mapping

The 22 recommendations map to the four AI RMF functions as follows (some recommendations map to multiple functions): GOVERN receives the most, reflecting the emphasis on foundational governance vocabulary. MEASURE and MAP address assessment and risk characterization. MANAGE covers operational controls. Key mappings:

Recommendation Group	Primary AI RMF Function	Key Subcategories
Recognize InAC (1, 3, 4)	GOVERN	1.1, 1.2 (legal/regulatory, trustworthy AI)
Enforce at Boundaries (5, 6, 8)	GOVERN + MANAGE	1.2, 1.4 (risk management); 1.3 (risk response)
Monitor Interior (2, 7, 16)	MEASURE	2.4, 2.7, 3.1 (production monitoring, security, risk tracking)
Secure Identity (17, 20)	MEASURE + GOVERN	2.7 (security evaluation); 1.1, 2.1 (legal, roles)
Governance Maturity (9–12)	GOVERN + MAP	1.3, 1.5, 1.6 (risk levels, monitoring, inventory); 1.5 (risk tolerance)
Threat Catalog (14, 15, 18)	MAP + MANAGE	4.1, 5.1 (risk mapping, impact); 1.3 (risk response)
SP 800-53 Controls (19)	GOVERN	1.1, 1.2 (legal/regulatory, trustworthy AI)
Industry Coordination (21, 22)	GOVERN	1.3, 4.3, 5.2 (activity levels, sharing, feedback)

CSF 2.0's Govern function parallels AI RMF Govern; recommendations 1, 5, 9–13, and 19–22 map to both, enabling unified governance implementation. (Full per-recommendation mapping: Technical Companion, Section 8.8.)

Related Work

The InAC formalization builds upon three decades of normative multi-agent systems research while addressing a case these frameworks were not designed for: agents whose norm compliance emerges from language model alignment rather than programmed logic.

Normative MAS. Shoham and Tennenholtz (1995) introduced social laws for artificial agent societies. Boella and van der Torre (2004) formalized regulative and constitutive norms with explicit compliance/violation decisions. Dignum (2004) developed organizational structures for norm enforcement. All assume agents have explicit norm-reasoning modules (BDI architectures, deontic logic). InAC differs fundamentally: norm compliance is emergent from alignment training, not programmed; norms resist formalization (Axiom 3); and the Subject-Enforcement Identity (E=S) means no external monitor exists — a property absent from all prior normative frameworks.

Social commitments. Singh (1999, 2021) models commitments as first-class social objects tracked through protocol mechanisms. InAC norms are implicit in natural language, interpreted

probabilistically, and carry no machine-verifiable metadata — which is why the confused deputy attack (Section 7.1) succeeds.

Trust theory. Castelfranchi and Falcone's (2010) cognitive trust model treats trust as a belief about another agent's competence. InAC's alignment function A(s) is structurally analogous but is a property of the agent's training, making it empirically measurable through benchmarks rather than subjectively assessed — positioning alignment as a security property for ATO documentation.

Institutional governance. Ostrom's (2005) IAD framework for governing commons provides the theoretical foundation for L5 self-governance. Her conditions for successful self-governance (monitoring, graduated sanctions, conflict resolution) map directly to L1-L4 as prerequisites. The "Premature Self-Governance" anti-pattern is the multi-agent analog of Ostrom's finding that self-governance fails without institutional prerequisites.

Electronic institutions. Artikis, Sergot, and Pitt (2005) developed runtime norm monitoring — the closest prior work to E_o. The key distinction: electronic institutions assume formally specified norms evaluated by an external monitor; InAC systems have natural language norms with no external monitor, making E_o not just useful but essential.

(Full Related Work discussion with detailed comparisons: Technical Companion, Section 8.9.)

---

9. Conclusion

9.1 Timeliness

The NIST/CAISI RFI represents a timely opportunity to establish foundational concepts for AI agent security guidance before patterns of deployment become entrenched in federal IT infrastructure. Current standards — NIST AI RMF, ISO/IEC 42001, OWASP Top 10 for Agentic Applications — address the deterministic outer shell of AI agent security (authentication, authorization, content filtering) but do not yet address the intrinsic enforcement interior. Guidance development now can establish assessment requirements before they must be retrofitted to operational systems.

9.2 Four Contributions for NIST Consideration

This submission proposes four contributions for NIST evaluation:

1. **InAC formal specification** (Section 4) as a candidate theoretical foundation for AI agent interior security. The four axioms, one empirical property, four theorems, and formal comparison table provide a basis for standards language about probabilistic, intrinsically-enforced controls. Empirical validation is planned for Q2 2026.

2. **Enforcement Location Principle** (Section 5) as a candidate reference framework for AI agent security architecture. The E_d/E_n/E_o taxonomy and placement requirements, consistent with the trust zone concepts in NIST SP 800-207 (Zero Trust Architecture), provide actionable guidance for architects and assessors.

3. **Governance Maturity Model** (Section 6) as a candidate federal self-assessment framework. The L0-L5 model with six dimensions and scoring rubrics could complement the NIST Cybersecurity Framework 2.0's Govern function for AI agent systems specifically.

4. **Threat taxonomy** (Section 7) as a candidate adversarial analysis framework. The 47-vector taxonomy across 6 categories, with estimated InAC resistance rates, provides a structured basis for AI agent threat modeling requirements.

9.3 Risks of Omission

Absent formal guidance addressing intrinsic access control, federal system security plans and ATO documentation will assess only deterministic boundary controls in AI agent systems, leaving interior behavioral governance without a compliance pathway. The threat taxonomy in Section 7 identifies attack categories — including prompt injection, confused deputy exploitation, and identity forgery — that are not captured by current FISMA documentation requirements.

9.4 Proposed Vocabulary

Beyond specific technical recommendations, this submission proposes vocabulary that NIST may wish to evaluate for adoption in AI agent security guidance:

- **InAC + ELP** as a more precise alternative to "guardrails" (which implies only boundary enforcement without naming the interior enforcement model).
- **Compliance probability C(s, n)** as a measurable alternative to "trust in the AI system."
- **E_o (observational enforcement)** as a required architectural component rather than optional "monitoring."
- **Alignment as a security property A(s)** as a bridge between AI safety vocabulary and security risk assessment.

Adoption of this vocabulary in NIST guidance would provide assessors across the federal enterprise with conceptual tools to evaluate the most important and least addressed dimension of AI agent security: the behavior governed by intrinsic access control between deterministic enforcement points.

---

10. Formal References

Access Control Foundations

- Anderson, J.P. (1972). "Computer Security Technology Planning Study." ESD-TR-73-51.
- Bell, D.E. and LaPadula, L.J. (1973). "Secure Computer Systems: Mathematical Foundations." MITRE Technical Report.
- Biba, K.J. (1977). "Integrity Considerations for Secure Computer Systems." MITRE Technical Report.
- Clark, D.D. and Wilson, D.R. (1987). "A Comparison of Commercial and Military Computer Security Policies." IEEE S&P.
- Sandhu, R. et al. (2000). "The NIST Model for Role-Based Access Control." NIST.
- Ferraiolo, D.F. et al. (2001). "Proposed NIST Standard for RBAC." ACM TISSEC.
- Hu, V.C. et al. (2014). "Guide to ABAC Definition and Considerations." NIST SP 800-162.

AI Security and Prompt Injection

- Perez, F. and Ribeiro, I. (2022). "Ignore Previous Prompt: Attack Techniques for Language Models." NeurIPS ML Safety Workshop. arXiv:2211.09527.
- Greshake, K. et al. (2023). "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection."
- OWASP (2025/2026). "Top 10 for Agentic Applications."

Formal Methods

- Rice, H.G. (1953). “Classes of Recursively Enumerable Sets and Their Decision Problems.” Trans. AMS.
- Waismann, F. (1945). “Verifiability.” Proc. Aristotelian Society.

NIST Standards Referenced

- NIST AI Risk Management Framework (AI RMF) 1.0, January 2023.
- NIST Cybersecurity Framework (CSF) 2.0, February 2024.
- NIST SP 800-53 Rev. 5, “Security and Privacy Controls for Federal Information Systems.”
- NIST SP 800-63-3, “Digital Identity Guidelines.”
- NIST SP 800-162, “Guide to ABAC Definition and Considerations.”
- NIST SP 800-207, “Zero Trust Architecture.”
- FIPS 199, “Standards for Security Categorization of Federal Information and Information Systems.”

Adversarial ML Frameworks

- MITRE ATLAS (Adversarial Threat Landscape for AI Systems).

Industry Standards and Frameworks

- EU AI Act (Regulation (EU) 2024/1689). Full application: August 2, 2026.
- Agentic AI Infrastructure Foundation (AAIF), Linux Foundation, December 2025.
- CISA (2026). “Principles for Secure AI Integration in Operational Technology.”

Normative Multi-Agent Systems and Institutional Governance

- Shoham, Y. and Tennenholtz, M. (1995). “On Social Laws for Artificial Agent Societies: Off-Line Design.” Artificial Intelligence.
- Boella, G. and van der Torre, L. (2004). “Regulative and Constitutive Norms in Normative Multiagent Systems.” KR-2004.
- Dignum, V. (2004). “A Model for Organizational Interaction.” PhD thesis, Utrecht University.
- Singh, M.P. (1999). “An Ontology for Commitments in Multiagent Systems.” AI and Law.
- Singh, M.P. (2021). “Maintenance of Social Commitments in Multiagent Systems.” AAAI-2021.
- Artikis, A., Sergot, M., and Pitt, J. (2005). “Implementing Norms in Electronic Institutions.” AAMAS-2005.
- Castelfranchi, C. and Falcone, R. (2010). “Trust Theory: A Socio-Cognitive and Computational Model.” Wiley.
- Ostrom, E. (2005). “Understanding Institutional Diversity.” Princeton University Press.

Research Sources (this submission)

The analytical findings in this submission derive from the following research components, available upon request:

- “Formal Specification of Intrinsic Access Control.” Multi-Agent Systems Security Research Program, February 2026.
- “Attack Taxonomy Against Intrinsic Access Control.” Multi-Agent Systems Security Research Program, February 2026.

- “Enforcement Location Principle: Formal Specification.” Multi-Agent Systems Security Research Program, February 2026.
- “Governance Maturity Model for Multi-Agent Systems.” Multi-Agent Systems Security Research Program, February 2026.
- “Industry Landscape Survey: Multi-Agent Governance.” Multi-Agent Systems Security Research Program, February 2026.

body { font-family: “Times New Roman”, serif; font-size: 11pt; margin: 1in; } table { border-collapse: collapse; width: 100%; font-size: 9pt; } th, td { border: 1px solid #ccc; padding: 4px; } h1 { font-size: 16pt; } h2 { font-size: 14pt; } h3 { font-size: 12pt; }