March 9, 2026

Response to the Center for AI Standards and Innovation (CAISI)'s Request for Information Regarding Security Considerations for Artificial Intelligence Agents

Docket ID No. NIST-2025-0035. Submitted Electronically.

Executive Summary

The central security challenge of AI agent systems is that language models process instructions and data in the same channel, with no reliable separation boundary. When these models are connected to real tools and real data, a successful attack does not produce a wrong answer; it produces unauthorized actions. This architecture is a feature, not a bug, and therefore requires comprehensive defenses.

Below, you'll find Dreadnode's response to the Request for Information Regarding Security Considerations for Artificial Intelligence Agents released by CAISI under the National Institute of Standards and Technology (NIST-2025-0035). This response makes three claims, substantiated across five sections covering threats, controls, evaluation, deployment, and policy. First, the most dangerous vulnerabilities in agentic systems are not implementation flaws; but rather inherent properties of systems that conflate instruction processing with data processing, delegate real-world authority to probabilistic models, and compose trust implicitly across agent boundaries. Second, the single highest-leverage gap across the ecosystem is the absence of identity and access management at the tool invocation layer. Most deployed agents operate with a flat set of credentials regardless of what context triggered a request, meaning every tool accessible to the agent is effectively accessible to every user who can interact with it. Third, the controls that will prove most durable are those that do not depend on the model behaving correctly: hard-coded action limits enforced outside the model, least-privilege tool access scoped to the originating principal, and behavioral monitoring that detects anomalous actions regardless of how the underlying attack was delivered.

The area where government action is both most urgent and most likely to be transformative is agent identity, authentication, and authorization standards. No individual company can establish universal agent identity norms. To the contrary, this is a coordination problem the government is uniquely positioned to solve. Federal procurement requirements mandating agent identity standards and require Policy-as-Code solutions would create the market demand to accelerate private-sector adoption, as FedRAMP and soon FedRAMP 20x continue to do for cloud security.

1. Security Threats, Risks, and Vulnerabilities Affecting AI Agent Systems

(a) What are the unique security threats, risks, or vulnerabilities currently affecting AI agent systems, distinct from those affecting traditional software systems?

Foundational Risks, Compounded

AI agent systems inherit the full threat landscape of traditional software, including unpatched CVEs, misconfigurations, exposed credentials, insufficient supply chain verification. These foundational risks are compounded by an entirely new class of threats on top. In practice, most agentic deployments have not adequately addressed even these foundational risks. For AI and ML workloads specifically, supply chain exposure is more acute than in traditional software: teams are pulling in model weights, fine-tuning datasets, inference frameworks, and plugin ecosystems where provenance and integrity are rarely verified. These are not solved problems, and they are not made easier by the additional complexity of the agentic layer.

The Agent as Attack Vector

What is unique to agentic AI is that the agent itself becomes the attack vector. Infrastructure can be fully hardened by patching CVEs, rotating credentials, and encrypting endpoints; yet, an adversary can still compromise the system by manipulating the natural language flowing through it. Prompt injection, both direct and indirect, allows an attacker to hijack an agent's goals and turn its own capabilities against the system it serves. This is not a vulnerability that can be patched. It is inherent to how language models process instructions and data in the same channel without a reliable separation boundary.

Non-Determinism and Verification Complexity

Unlike traditional software systems, AI agent systems are inherently non-deterministic. The same input may not produce identical outputs, which makes reproducibility, regression testing, and verification of security fixes significantly more complex. In traditional software, vulnerabilities can typically be patched with deterministic code changes and validated through repeatable test cases. In contrast, AI agent mitigations may reduce, but not fully eliminate, undesired behaviors. Emergent behaviors arising from the combination of reasoning autonomy and external tool invocation are difficult to model, sandbox, or fully constrain using conventional security controls.

Unauthorized Action, Not Just Wrong Answers

What makes prompt injection uniquely dangerous in agentic contexts is that these agents are connected to real tools and real data. A successful injection does not produce a wrong answer; it produces unauthorized actions. Attackers can weaponize the agent's own tool access to execute arbitrary commands, chain individually legitimate operations into unauthorized sequences under the agent's own credentials, poison conversation memory to create persistent backdoors, and exfiltrate sensitive data through the conversational channel itself rather than through any network breach.

The Identity and Access Management (IAM) Gap at the Tool Layer

A critical structural gap in most current agentic deployments is the absence of identity and access management at the tool invocation layer. When an agent invokes a tool, there is typically no validation of whether the originating user, session, or upstream agent has the privileges required for that specific action. The agent operates with a single flat set of credentials regardless of what context triggered the request. There is no mechanism to enforce role-based tool permissions—i.e. restricting a customer-facing agent to read-only access—or to differentiate privileges based on whether a call originates from external input or an internal workflow. Every tool accessible to the agent is effectively accessible to every user who can interact with it.

Inadequacy of Traditional Assurance Methods

Traditional security assurance methods, including static analysis, deterministic unit testing, and formal verification, are less effective in AI agent systems. The field requires new evaluation methodologies focused on behavioral risk, probabilistic outcomes, and adversarial robustness. Systematically probing agentic systems with adversarial inputs across different injection surfaces, tool access patterns, and multi-agent trust chains is currently the most practical method for mapping real-world exposure. These evaluations must be built into the development lifecycle, not treated as a post-deployment activity.

The foundational mitigation posture requires enforcing tool-level constraints architecturally rather than through prompting, hard-coding limits on what actions agents can take regardless of instructions, implementing IAM and Role-Based Access Control (RBAC) at the tool invocation layer, and designing multi-agent trust boundaries enforced by the system rather than by the agents themselves.

(b) How do security threats, risks, or vulnerabilities vary by model capability, agent scaffold software, tool use, deployment method, hosting context, and use case?

These variables are not independent; they compound. A change in any one of them can materially alter the system's threat model.

Model Capability

Model capability is the single most determinative variable. Even when architecture, tools, and system prompts are identical, the underlying model determines how resistant the system is to adversarial manipulation. Frontier models with stronger safety training tend to hold their instructions under adversarial pressure; smaller or less-aligned models will follow injected instructions with little resistance. The same attack that one model refuses, another executes without hesitation. Model selection is therefore as much a security decision as an architectural one. Organizations must evaluate models for adversarial robustness, not only task performance, and must re-evaluate their full security posture whenever a model is swapped or updated.

Agent Scaffold and Tool Access

The agent scaffold and tool configuration determine blast radius. A single agent with read-only access to a narrow data source is a limited target. A multi-agent pipeline with shared memory, delegation capabilities, and write-capable or execution-capable tools presents an enormous attack surface. Most orchestration frameworks default to maximum model flexibility, which is equivalent to maximum attack surface. In most deployments there is no enforcement layer between the model deciding to invoke a tool and the tool executing. The distinction between read-only and write-capable tools is particularly consequential: the former primarily creates data exposure risk; the latter introduces integrity and availability risks.

Deployment Context and Hosting Environment

External deployments accepting anonymous input face fundamentally different threats than internal deployments behind authentication layers, though internal deployment does not eliminate injection risk, since data-level injection can occur through documents, emails, and tool outputs processed by the agent. Cloud hosting enables centralized monitoring but routes data through third-party infrastructure. Edge deployments make real-time monitoring impractical and limit available security tooling. The use case ultimately determines whether a given vulnerability is acceptable: the same exposure that is minor in a meeting-summarization agent is catastrophic in a financial transaction or clinical decision system.

Integration Depth and Enterprise Embedding

The more deeply an agent is integrated into identity systems, data stores, and operational workflows, the greater the potential impact of compromise or misalignment. Integration depth is a multiplicative factor on all other risk variables.

(c) To what extent are security threats creating barriers to wider adoption or use of AI agent systems?

At present, security risks do not appear to be materially slowing overall AI agent adoption in aggregate. Market and competitive pressures are driving rapid deployment across sectors, frequently outpacing the maturation of formal security controls. The barrier is not that organizations are unaware of risks; it is that they cannot quantify the maximum downside.

When an agent has access to internal tools, data, and connected systems, the worst-case outcome is not a wrong answer. Instead, it is full compromise: exfiltrated intellectual property, unauthorized transactions, persistent backdoors in memory, or cascading failures across connected agents and services. Most organizations can tolerate known risk. With agentic systems today, the ceiling of potential damage is unclear because the blast radius of a successful attack depends on what the agent can reach, and most deployments have not mapped that.

AI red teaming is the most practical intervention. Systematically probing agentic systems with adversarial attacks across different injection surfaces, tool access patterns, and multi-agent trust

chains produces concrete visibility into what can actually go wrong and how severe it can be. Until organizations have that visibility, security teams will continue to block or heavily restrict agentic deployments in high-value environments. Security maturity should be understood as a critical enabler of sustainable adoption, not as a gating factor opposed to it.

(d) How have these threats changed over time, and how are they likely to evolve in the future?

The impact of these threats has grown in direct proportion to what AI systems are permitted to do. When models could only generate text, the worst outcome was harmful content. When they were connected to retrieval systems, the risk expanded to data leakage. Now that they can take autonomous actions, execute tools, manage workflows, and delegate to other agents, a successful attack can result in unauthorized transactions, data exfiltration, persistent system compromise, and cascading failures across connected services. Each new capability granted to an agent is simultaneously a new attack surface.

The trajectory of risk is not linear. As agents gain persistent memory, deeper enterprise integration, and the ability to operate across organizational boundaries with reduced human oversight, risks compound in kind. Attacks that are session-scoped today become persistent when agents retain state. Single-agent compromises become lateral movement paths when agents delegate to each other. The pool of potential attackers grows as agents move from internal tools to customer-facing autonomous systems exposed to the open internet.

Looking further ahead, the field should expect adversaries to use AI agents to attack other AI agents, generating adaptive, multi-turn attack strategies at a speed and scale that manual red teaming cannot match. The security challenge will shift from defending individual agents to securing entire ecosystems of agents that interact, transact, and establish trust autonomously. The controls that will age best are those that do not depend on the model behaving correctly: architectural enforcement, least-privilege access, and behavioral monitoring that detects anomalous actions regardless of how the underlying attack was delivered.

(e) What unique security threats affect multi-agent systems, distinct from those affecting singular AI agent systems?

Inter-Agent Trust as an Attack Surface

The core risk unique to multi-agent systems is that compromising one agent provides a foothold to reach every agent connected to it through delegation, shared memory, or message passing. In a single-agent system, blast radius is bounded by what that agent can access. In a multi-agent system, agents pass instructions to each other, delegate tasks, share context, and act on each other's outputs, none of which is cryptographically authenticated or verified. When one agent instructs another to perform an action, the receiving agent has no reliable mechanism to verify whether that instruction originated from a legitimate workflow or from an attacker who compromised the sending agent.

Privilege Escalation Without Vulnerability Exploitation

This architecture enables privilege escalation without exploiting any software vulnerability. An attacker compromises a low-privilege customer-facing agent and uses it to send crafted instructions to a higher-privilege backend agent with access to sensitive tools and data. The backend agent trusts its peer and executes the request, achieving privilege escalation through manipulation of inter-agent trust that cannot be cryptographically verified at the semantic layer. This pattern of compromising a weak entry point and pivoting to higher-value targets through trust relationships mirrors lateral movement in traditional network attacks, but occurs entirely in the semantic layer where conventional security tooling has no visibility.

Shared State Poisoning

Multi-agent systems also introduce shared state risks absent from single-agent deployments. When agents share memory, context stores, or conversation history, a single injection into a shared state becomes a persistent, system-wide backdoor affecting every agent that reads from it. The difficulty of tracing which agent introduced a compromise, and which downstream agents and actions were affected, makes incident response and forensics significantly more complex than in single-agent architectures.

Securing multi-agent systems requires cryptographic or token-based inter-agent authentication, sanitization of all inter-agent message content analogous to input validation in web applications, and trust boundaries enforced architecturally by the system rather than negotiated in natural language between agents.

2. Security Practices for AI Agent Systems

(a) What technical controls, processes, and practices could ensure or improve the security of AI agent systems? What is the maturity of these methods?

The current industry default is to rely on model-level safety training as the primary security defense and treat everything else as optional. This is inadequate. Model-level controls set the floor, but the controls with the greatest impact and the greatest durability are at the agent system level, specifically architectural constraints that limit what an agent can do regardless of what the model outputs. Security for agentic systems must be layered across the model, the agent system, and human oversight. No single layer is sufficient. Controls at each layer must be architectural and enforced, not advisory and prompt-based.

Model-Level Controls

The foundation model's robustness to adversarial manipulation sets the floor for the entire system's security posture. Training-time interventions, including safety fine-tuning and adversarial robustness training, help models maintain intended behavior under injection pressure. Inference-time measures include input classifiers that detect and filter known injection patterns before they

reach the model, and output classifiers that flag responses where the model appears to be deviating from its intended behavior. Current safety training significantly reduces attack success rates on frontier models relative to base or smaller models, but no model today is reliably immune to adaptive multi-turn attacks. The critical gap in practice is that most deployers rely entirely on the model provider's default safety training without additional hardening for their specific deployment context. A model evaluated as robust in a customer service scenario may behave very differently when connected to financial transaction tools or code execution environments. Model-level evaluations must be deployment-specific: organizations should adversarially test the model against the actual tools, data sources, and interaction surfaces it will encounter in production, and must re-evaluate whenever the model is updated or swapped. Maturity here is moderate in research and inconsistent in practice.

Agent System-Level Controls

This is where the most impactful and most neglected controls reside. The governing principle is that security constraints must be enforced in code outside the model, not expressed as instructions inside the system prompt. A system prompt instruction telling the model "never transfer more than $10,000" is not a security control; it is a suggestion that will fail under sustained adversarial pressure. A hard-coded transaction cap in the tool execution layer that returns an error regardless of what the model requests is a security control. This distinction, between prompt-based guidance and architectural enforcement, is the single most important design decision in agent security, and most current deployments get it wrong. Tool-level guardrails must enforce hard limits on what actions an agent can take regardless of what the model outputs, including transaction caps, restricted tool parameters, blocklists for sensitive operations, and mandatory confirmation steps for high-risk actions. IAM and RBAC at the tool invocation layer should scope tool permissions to the originating user or session, not grant a flat set of capabilities to the agent process. Input and output sanitization between agents in multi-agent systems should strip or flag embedded instructions in inter-agent messages, analogous to how web applications sanitize user input against cross-site scripting. Continuous monitoring of agent behavior, including tool call patterns, delegation chains, and anomalous action sequences, provides runtime detection of compromise. The maturity of these practices is low. Most orchestration frameworks do not implement them by default, and most deployments have not added them independently. The structural reason is that current frameworks prioritize developer experience and model flexibility, making the insecure path easier than the secure one.

Human Oversight Controls

For high-consequence actions, a human-in-the-loop approval gate must be enforced by the system architecture and cannot be bypassed by the agent regardless of its instructions. This applies to actions such as financial transactions above defined thresholds, creation or modification of user accounts, access to sensitive data stores, and execution of code in production environments. If the model can choose to circumvent an approval step, it will choose to do so under adversarial pressure. Human oversight is the most conceptually mature control category but is frequently

implemented poorly in practice: either as a rubber-stamp approval that adds friction without security value, or as monitoring that generates alerts no one reviews. The failure mode is not missing oversight; it is performative oversight. The design principle that separates effective oversight from theater is decision quality: the human must receive sufficient context to make a genuinely informed judgment, not merely a yes/no prompt they will habituate to clicking through. An approval interface that presents "Agent wants to execute action. Approve?" without showing the full chain of reasoning, the data accessed, or the downstream consequences of the action is security theater regardless of how many approvals it collects. Beyond per-action approvals, organizations need comprehensive audit logging of all agent actions with sufficient detail to reconstruct decision chains, regular review of agent behavioral patterns to detect drift, and kill switches capable of immediately revoking tool access when anomalies are detected.

Evaluations as a Core Security Practice

Rigorous, domain-informed evaluation frameworks are foundational to AI agent security. Continuous red-teaming, adversarial testing, and scenario-based assessments are currently among the most effective methods for identifying prompt injection susceptibility, tool misuse pathways, and systemic weaknesses. Evaluation must be integrated into the development lifecycle, not treated as a release-gate activity, and must be re-run after any material change to the model, scaffold, tools, or deployment context. The critical infrastructure gap is the absence of standardized evaluation suites and adversarial benchmarks specific to agentic threat models. Today, every organization building agents must construct its own evaluation methodology from scratch, with no common baseline against which to measure progress or compare systems. This is precisely the kind of standards infrastructure NIST is positioned to support, and it would have an outsized impact on ecosystem-wide security maturity. A shared evaluation framework for agentic systems, covering prompt injection resistance, tool misuse pathways, multi-agent trust exploitation, and behavioral anomaly detection, would enable collective learning across the field in a way that proprietary, ad hoc testing cannot.

(b) To what degree could the effectiveness of technical controls vary with changes to model capability, scaffold, tool use, deployment method, or multi-agent use?

The effectiveness of every control is context-dependent. No control works uniformly across all configurations, and the right control set must be matched to the specific combination of model, scaffold, tools, deployment, and use case, and re-evaluated whenever any of those variables change.

Model-level controls such as safety fine-tuning are most effective on frontier models and significantly less effective on smaller or open-weight models. A prompt injection defense validated on one model may fail entirely on another, meaning model selection decisions are implicitly security decisions and require independent adversarial evaluation. System-level controls such as tool guardrails and IAM are the most stable across configurations because they operate outside the model, but their effectiveness depends on the scaffold's architecture. Frameworks that allow

models to call tools directly with no intermediary enforcement layer make guardrails harder to enforce than those with a structured tool execution tier. In multi-agent systems, system-level controls become both more important and more difficult to implement, since every agent-to-agent boundary requires its own sanitization and access control. Human oversight controls degrade rapidly as agent autonomy and operational tempo increase. An approval step practical for an internal agent handling five requests per hour is unworkable for a customer-facing agent handling thousands.

(c) How might technical controls need to change in response to the likely future evolution of AI agent capabilities or of the threats facing them?

Controls that depend on recognizing known attack patterns will become less effective as adversaries use AI to generate novel, adaptive attacks at scale. Static input filters and rule-based detection will not keep pace with algorithmically generated multi-turn attacks optimized to evade specific defenses. The required shift is toward behavioral monitoring that detects anomalous agent actions regardless of the attack's delivery mechanism, and toward architectural enforcement that limits what an agent can do even if the underlying model is fully compromised.

As agents become more autonomous and operate with less human oversight, the intervention window shrinks. Automated circuit breakers, rate limits on consequential actions, and mandatory cooling-off periods for irreversible operations will become essential rather than optional safeguards. The controls that will age best are those that do not depend on the model behaving correctly. Policy-as-code approaches, where security constraints are expressed as machine-enforceable rules evaluated independently of model outputs, represent the most durable defensive architecture as capabilities advance.

(d) What are the methods, risks, and considerations relevant for patching or updating AI agent systems throughout the lifecycle?

Patching an agentic system is fundamentally different from patching traditional software because changing any component can alter the agent's behavior in ways that are difficult to predict. Replacing or updating the underlying model can modify how the agent responds to adversarial inputs, invalidate existing safety guardrails tuned to the previous model's behavior, or introduce new vulnerabilities not present in the prior version. Unlike a traditional software patch where deterministic inputs and outputs can be tested, an agent's behavior is probabilistic and context-dependent, requiring adversarial evaluation rather than functional regression testing alone to establish that the update is safe.

Updates to system prompts, tool configurations, or agent-to-agent communication patterns carry equivalent risk. A prompt change intended to improve safety in one scenario can degrade it in another. Adding a new tool immediately expands the attack surface; removing a tool can break downstream agents that depended on it. Every change to an agentic system, whether a model swap, a prompt update, or a tool addition, must be treated as a security-relevant event requiring re-

evaluation through adversarial testing before deployment. The traditional patch-test-deploy lifecycle does not account for the emergent and adversarial dimensions of agentic behavior.

(e) Which cybersecurity guidelines, frameworks, and best practices are most relevant to the security of AI agent systems?

Existing frameworks provide a useful starting point but none fully address the distinctive risks of agentic systems. The NIST AI Risk Management Framework and the NIST Cybersecurity Framework provide foundational governance and risk management structures. The OWASP Top 10 for LLM Applications and Agentic Applications covers prompt injection and related model-level and agentic risks. MITRE ATLAS catalogs adversarial techniques against AI systems. These should all be applied.

The gap is that none of these frameworks adequately addresses the agentic layer: tool-level access control, inter-agent trust, delegation chain security, shared state integrity, and the compound risks that emerge when autonomous agents interact with each other and with real-world systems. What is needed is guidance that bridges traditional cybersecurity practices such as least privilege, network segmentation, and zero trust with the new realities of language-model-based autonomy, where the threat is behavioral rather than structural. Frameworks should evolve to treat agent tool access with the same rigor applied to API access management, inter-agent communication with the same scrutiny applied to network traffic, and agent behavioral drift with the same seriousness as configuration drift in traditional infrastructure.

Adoption and Impediments

Adoption of existing frameworks by AI agent developers and deployers is low. Teams building agentic systems are generally aware of prompt injection as a concept but rarely implement layered systematic defenses against it. Tool-level access controls, inter-agent message sanitization, and behavioral monitoring are almost never present in early deployments. Security frameworks are well-known within security organizations but have limited reach into AI engineering teams, who typically treat security as something to be added after initial functionality is established rather than built in from the start.

The most consequential misconception is that system prompt instructions constitute security controls. Instructions telling the model never to perform a specific action are insufficient protection: they fail under sustained adversarial pressure, and any system whose security depends on a prompt remaining secret is insecure by design. A secondary impediment is that most agentic development frameworks prioritize developer experience and model flexibility over security, making it structurally easier to build an insecure agent than a secure one.

3. Assessing the Security of AI Agent Systems

(a) What methods could be used during AI agent systems development to anticipate, identify, and assess security threats, risks, or vulnerabilities?

The most effective pre-deployment method is structured AI red teaming, where adversarial agents systematically probe the system using multi-turn attack strategies across different injection surfaces, tool access patterns, and agent delegation chains. Teams should develop threat scenarios specific to their agent's capabilities, identifying worst-case outcomes for each tool, each data source, and each agent-to-agent trust relationship. Automated red teaming using AI-driven attack generation scales this process beyond what manual testing allows and continuously generates adaptive attacks that evolve against the target system's defenses.

Complementary methods include static analysis of the agent's tool graph and permission model to identify overprivileged configurations before deployment; scenario-based tabletop exercises that walk through attack paths specific to the agent's architecture; and supply chain audits covering model provenance, training data integrity, and third-party tool dependencies. These audits should be standard before any agent reaches production, not optional enhancements.

Post-Deployment Detection

Post-deployment detection requires continuous monitoring of agent behavior rather than only infrastructure metrics. This means logging every tool invocation with full context—what triggered it, which user or agent initiated it, and what data was accessed—and monitoring for anomalous action sequences, unexpected delegation patterns, and behavioral drift from the agent's established baseline. When incidents occur, post-incident analysis must reconstruct the complete chain of events across all agents involved to understand not only what happened but why existing controls failed to detect it. Findings must feed directly back into the red teaming process and update guardrails accordingly. Standard security information and event management tooling is a necessary but insufficient foundation: it must be extended to cover natural language interaction logs, tool call sequences, and agent-to-agent message flows that traditional Security Information and Event Management tools were not designed to parse.

Maturity of Detection Methods

AI red teaming is maturing rapidly, with commercial platforms and open-source frameworks now available, but most organizations have not yet integrated it into their development lifecycle. Automated attack generation is active in research but early in applied use. Behavioral monitoring for agents is nascent, with most teams relying on basic logging rather than purpose-built anomaly detection. Supply chain security for AI components, specifically auditing model provenance and training data integrity, is the least mature area, with very few organizations conducting it in any systematic way.

(b) How could the security of a particular AI agent system be assessed, and what information helps with that assessment?

Not every threat applies to every agent. Assessment should begin with mapping the agent's specific risk profile: what tools it can access, what data it can reach, who can interact with it, whether it delegates to other agents, and what the worst-case outcome would be if it were fully compromised. From that profile, targeted red teaming against the highest-risk combinations of tools, data, and interaction surfaces produces a realistic picture of actual vulnerability rather than theoretical risk.

The information that aids assessment most directly is a clear inventory of tool permissions, delegation relationships, data sensitivity classifications, and exposure scope (internal vs. external users). Without this inventory, assessment is guesswork. Organizations should treat this inventory as a first-order security deliverable, not a documentation afterthought.

(c) What documentation or data from upstream AI model developers might aid downstream providers in assessing and managing security threats?

Downstream deployers require documentation covering: the safety training the model received and the adversarial evaluations performed; known failure modes and conditions under which the model is most susceptible to instruction override; how the model behaves when given conflicting or contradictory instructions; susceptibility to multi-turn persuasion; and whether the model has been evaluated for tool-use safety scenarios relevant to agentic deployment. For agentic applications specifically, general capability cards are insufficient. What is needed is adversarial characterization of model behavior under the specific threat conditions that arise in autonomous tool-using contexts.

Open-Weight vs. Closed-Weight Models

Open-weight models allow downstream teams to run independent adversarial evaluations and fine-tune for specific safety requirements, but carry higher risk because they can be modified and redistributed without safety guarantees and without ongoing provider-level safety updates. Closed-weight models provide less visibility into model behavior but typically include more structured safety documentation and continuous safety maintenance. Neither model class is inherently more secure. The key difference is the distribution of responsibility for safety evaluation between provider and deployer. In both cases, the downstream deployer retains responsibility for evaluating the model's behavior in their specific deployment context.

Responsible Disclosure

Vulnerability disclosures for agentic systems require the same responsible disclosure norms that govern traditional software: findings reported privately to the vendor first, mitigations developed, and public disclosure timed to give affected parties the opportunity to respond. Disclosures that could create new vulnerabilities include: internal safety classifier thresholds, specific fine-tuning

techniques used for safety alignment, exact architectures of guardrail systems, and working exploit chains combining tool abuse, memory poisoning, and privilege escalation across multi-agent systems. The system prompt itself is not a meaningful disclosure risk; any system whose security depends on prompt confidentiality is already insecure by design.

(d) What is the state of practice for user-facing documentation of AI agent systems that supports secure deployment?

Current documentation practices are poor. Most model providers publish general safety guidelines and usage policies, but few provide deployment-specific security documentation guiding downstream developers on secure configuration for their specific use case. Documentation rarely covers which tools are safe to expose in which contexts, what trust boundaries to enforce in multi-agent deployments, what monitoring is necessary and sufficient, or what adversarial scenarios have been tested and with what results. The consequence is that every deployer is reconstructing agent security from first principles, frequently learning through incidents rather than through structured guidance. Standardized security deployment guides, analogous to the operating system and database hardening guides that exist for traditional infrastructure, would materially improve baseline security across the ecosystem.

4. Limiting, Modifying, and Monitoring Deployment Environments

(a) In what manner and by what technical means could the access to or extent of an AI agent system's deployment environment be constrained?

Constraining an agentic deployment environment requires controls at multiple layers, all enforced architecturally rather than through model instructions:

- Network-level: Egress filtering, firewall rules, VLANs, and air-gapping where appropriate limit the external resources an agent can reach regardless of its instructions.
- Process and tool access: Fine-grained, capability-based permissions that distinguish read, write, and execution access; hard-coded blocklists for sensitive operations; rate limits on consequential tool invocations.
- Credential and identity management: Principle of least privilege applied at the tool layer; short-lived credentials with automated rotation; agent identities distinct from human user identities; IAM scoped to the requesting principal rather than the agent process.
- Filesystem and storage: Containerized execution environments; segregated data stores; copy-on-write filesystems that preserve rollback capability.
- Observability and runtime: Comprehensive audit logging of all agent actions; classifier-based anomaly detection for non-deterministic behavior patterns; tripwires for access to sensitive resources; kill switches with immediate tool access revocation.

The governing principle is that control must be enforced outside the model and cannot be delegated to the model's own judgment about what is permissible.

(b) How could virtual or physical environments be modified to mitigate security threats? What is the state of rollback and undo capability for agentic systems?

Virtual environment modifications include copy-on-write filesystems and throwaway sandbox environments that enable state recovery, deception environments to detect and study attack behavior, and software-defined networking with micro-segmentation to isolate agent execution. Physical environment modifications for AI-connected operational technology include interlocks and dead-man switches, geofencing, and segregated operational technology networks.

State of Rollback Implementation

Research on rollback and undo capability substantially outpaces practical application. At the database and storage layer, mature mechanisms exist and are widely deployed. These mechanisms include Atomicity, Consistency, Isolation, Durability or ACID transactions, multi-version concurrency control, write-ahead logging, and event sourcing architectures all provide well-understood rollback capability. At the filesystem and infrastructure layer, git-based workspaces, ZFS and Btrfs snapshots, infrastructure-as-code with state management, and container snapshots provide similarly mature options.

At the application and API layer, where the gap between research and deployment is largest, compensating transactions (the Sagas pattern) define explicit semantic undo operations for each agent action, but designing comprehensive compensating actions for all possible agent operations in open-ended systems remains largely unsolved. Agent frameworks such as LangGraph provide checkpointing of agent state that enables restart from a prior checkpoint, but this handles agent state rollback rather than reversing external effects already executed. Speculative execution with approval gates, where the agent plans and simulates an action sequence before execution, prevents many rollback needs but does not address post-hoc recovery.

Reversibility-aware planning, where agents evaluate the reversibility of candidate actions before executing them and prefer reversible options, appears in safety-oriented agent research but is not standard in deployed systems. This represents a critical gap: for high-consequence deployments, the disposition to prefer reversible actions and escalate for human review before irreversible ones should be an architectural requirement, not an optional model behavior.

(c) What is the state of managing risks associated with interactions between AI agent systems and counterparties?

Interactions with Humans Not Using the System Directly

Risk awareness in this area is relatively high, but practices remain unsettled. When AI agents interact with third parties such as customers, counterparties in negotiations, and members of the public, risks span deception, manipulation, and legal liability. Emerging practices include disclosure requirements (informing humans they are interacting with an AI), scope-of-authority constraints, and human-in-the-loop approval for high-stakes decisions, but adoption is inconsistent. A robust approach requires explicit delegation and consent frameworks that define

and bound the agent's authority with respect to third parties, analogous to established protocols requiring explicit authorization for specific financial transactions. Maturity: low to moderate.

Interactions with Digital Resources

This is the area of most rapid security practice development, partly because it maps onto familiar application security paradigms. Real-world incidents have demonstrated the risks: hidden prompts have turned enterprise copilots into exfiltration engines, and agents have exploited legitimate tools to produce destructive outputs. Security researchers recommend extending controls across the full agent interaction chain, including prompts, retrieval steps, tool calls, and outputs, and validating, sanitizing, and assigning trust levels to all external content before agents ingest or act on it. The OWASP Top 10 for Agentic Applications provides the most comprehensive practitioner-oriented framework. Unlike traditional software supply chains, agentic ecosystems compose capabilities at runtime, loading external tools and agent personas dynamically, creating live supply chain exposure to poisoned prompt templates, tool descriptor injection, and malicious tool server compromise. Maturity: moderate, with attack surface expanding faster than defenses.

Interactions with Operational Technology and IoT

This is among the least mature areas for AI agent security, despite the broader field of OT and IoT security being well-established. AI agents introduce fundamentally new threat models into environments designed around deterministic controls and long asset lifecycles. OT assets built to last 10 to 15 years are typically insecure by design: patches are infrequent, and standard IT security tools cannot read proprietary OT protocols or baseline normal behavior. The convergence of AI agency with physical systems creates attack surfaces that existing frameworks were not designed to address. Emerging practices include zero-trust microsegmentation for isolating AI-connected industrial assets and AI-driven behavioral baselining for OT device networks. Adoption is limited. Maturity: low.

Interactions with Authentication and Network-Level Access

This area is receiving urgent attention driven by real incidents. AI agents require distinct authentication approaches including cryptographic attestation, hardware-backed key storage for service accounts, automated credential rotation, and integration with enterprise identity providers via standard federation protocols. The OWASP Agentic Security framework identifies identity and privilege abuse as a top-tier risk, where leaked credentials allow agents to operate far beyond their intended scope. The dynamic, runtime nature of agentic systems, where agents may request new permissions, chain tool calls, or generate and execute code, makes static access control models insufficient. Maturity: moderate, with significant implementation gaps.

Interactions with Other AI Agent Systems

This is the least mature and most concerning area. The rapid proliferation of multi-agent systems is creating interaction dynamics that are poorly characterized and almost entirely without

production-grade security controls. There are no widely adopted standards for agent-to-agent authentication, no established norms for how agents verify each other's identity or authority, and limited tooling for monitoring inter-agent communications. The Cooperative AI Foundation's taxonomy identifies three key failure modes: miscoordination (agents working at cross-purposes), conflict (agents competing in ways producing harmful outcomes), and collusion (agents developing cooperative strategies against their principals' interests). Specific deployed risks include agent communication poisoning, cascade failures from single-agent compromise propagating through a network, and emergent behaviors in multi-agent environments not anticipated by any individual agent's designers. Maturity: very low. This is largely an open research problem.

(d) What methods could be used to monitor deployment environments for security threats, risks, or vulnerabilities?

Available Monitoring Methods

Effective monitoring of agentic deployment environments requires capabilities beyond traditional application performance management. Key methods include: agentic observability and tracing, which captures reasoning traces, tool calls, data access events, and output evaluations in real time; behavioral anomaly detection, which identifies unusual access patterns, volume anomalies, off-hours activity, and boundary-testing behavior by mapping agent activities to adversarial technique frameworks; identity and access monitoring, which inventories all deployed agents, maps their connections, and continuously verifies that permissions align with intended scope; guardrail validation and content safety monitoring, which filters inputs and outputs and enforces prompt shield policies; and safeguard agents that monitor and constrain the behavior of production agents. OpenTelemetry's GenAI Special Interest Group is working to standardize agent telemetry collection across frameworks, providing a foundation for interoperable monitoring infrastructure.

Challenges to Deploying Traditional Monitoring

Traditional monitoring assumes deterministic systems with predictable behaviors. AI agents violate this assumption in ways that are fundamental rather than incidental. Traditional software fails loudly, throwing exceptions and returning error codes. AI agents fail silently: they confidently produce wrong outputs, leak data, or consume resources without generating observable errors. The same input can produce different outputs across runs, making anomaly detection against a stable baseline difficult to define. Perimeter defenses cannot inspect opaque model reasoning; static access control lists fail when agents dynamically acquire permissions; and signature-based threat detection misses adversarial inputs crafted to manipulate model reasoning. Additionally, the fragmentation of the agent framework landscape across LangChain, AutoGen, CrewAI, Semantic Kernel, and others means there is no consistent telemetry format, making unified security visibility across a heterogeneous agent estate difficult to achieve.

Legal and Privacy Considerations

Comprehensive security monitoring of agent systems requires logging the content agents process, which frequently includes personal data, health records, financial information, or confidential business data. This creates direct tension with data protection obligations. Under GDPR, monitoring logs that capture user interactions may themselves constitute personal data processing requiring a lawful basis. Conflicting retention obligations arise between GDPR's data minimization requirements and AI accountability obligations requiring comprehensive audit trails. Memory governance adds further complexity: short-lived working memory has a different risk and compliance profile than long-lived state stores, requiring distinct retention, access control, and deletion policies. Organizations must design monitoring architectures that achieve the observability required for security while maintaining compliance. This requires deliberate architectural investment, not ad hoc logging.

Maturity of Monitoring Methods

Standardization is nascent but accelerating. Agent observability tooling is at an early commercial stage: 2025 saw production deployments outpace observability practices, with teams lacking visibility into model degradation, cost anomalies, and non-deterministic failures. Behavioral monitoring for agents draws on mature machine learning security research but is only beginning to be applied to agentic systems specifically. Multi-agent monitoring, covering interactions between agents from different organizations or frameworks, remains a largely open research problem with no production-grade solutions at scale.

(e) Are current AI agent systems widely deployed on the open internet? How could traffic volume be tracked over time?

AI agent systems are now widely deployed on the open internet, and their presence is growing rapidly. Automated bot traffic surpassed human-generated traffic for the first time in a decade in 2024, a shift substantially attributable to the rise of AI agents. AI bot traffic across major Content Delivery Networks (CDNs) has grown by multiples in 2025, encompassing three distinct categories: training bots gathering content for model development; search and retrieval bots supporting AI-powered search; and user-action bots visiting sites as part of user-directed tasks; this last category, the most agent-like, is growing fastest. The boundary between bounded enterprise deployments and open-internet exposure is dissolving: even nominally internal agents frequently browse arbitrary websites, process external emails, and interact with third-party APIs.

Current tracking methods, including CDN-level traffic analysis, behavioral fingerprinting, robots.txt compliance monitoring, and emerging agent identity verification protocols, provide useful aggregate signals but are insufficient to fully characterize the scope and nature of agent activity. Fundamental challenges include the difficulty of identifying agents that do not self-identify or that impersonate human browsers, the absence of a unified measurement standard across CDN providers, and the invisibility of agent-to-agent traffic that does not traverse the public web in ways existing infrastructure can capture. Improved tracking would require standardized agent

identification headers, cross-provider measurement consortia, and registry-based approaches for agents operating in commerce and other high-consequence domains.

5. Additional Considerations

(a) What methods, guidelines, resources, or tools would aid rapid adoption of security practices and promote security innovation for AI agent systems?

Government-Led Standards and Reference Architectures

The NIST AI Agent Standards Initiative and the NCCoE AI Agent Identity and Authorization Concept Paper represent the most actionable government contributions to date. The NCCoE's approach of building working demonstrations with commercially available technologies in lab environments, showing implementation rather than prescribing it, is the right model and should be resourced to produce outputs within months. What would further accelerate adoption: sector-specific implementation guides translating general frameworks into concrete guidance for healthcare, financial services, critical infrastructure, and government; a NIST Special Publication specifically addressing AI agent security at the specificity practitioners need (tool misuse, memory poisoning, inter-agent communication attacks); and standardized security maturity models enabling organizations to benchmark their agent security posture systematically over time.

Community-Driven Frameworks and Threat Taxonomies

The OWASP Top 10 for Agentic Applications, developed with extensive industry collaboration, is currently the most widely adopted and operationally actionable community framework. Its strength is distilling complex threats into an accessible, prioritized format that security teams can apply immediately. The Coalition for Secure AI's MCP Security White Paper addresses protocol-level security threats across the Model Context Protocol. What would further aid adoption: hands-on training environments and capture-the-flag platforms that give security practitioners experience with agent-specific attacks; threat modeling templates pre-built for common agent architectures; and a continuously updated vulnerability database for agent systems analogous to CVE for traditional software, enabling collective ecosystem learning from incidents.

Identity and Authorization Infrastructure

Identity management is the single highest-leverage area for accelerating security adoption. The majority of organizations still treat agents as extensions of human users or as generic service accounts rather than as independent, identity-bearing entities. Reference implementations for agent identity using existing enterprise IAM infrastructure, built on OAuth 2.0, SPIFFE/SPIRE open source projects, or workload identity federation, would lower the adoption barrier substantially. The ecosystem needs open, vendor-neutral standards for agent-to-agent authentication: a security layer for inter-agent communication analogous to Transport Layer Security for network communication, easy to adopt and interoperable across vendors and frameworks.

Secure-by-Design Tooling

The most impactful structural change to the ecosystem would be agent development frameworks that ship with secure defaults rather than requiring developers to opt into security controls. Currently, most frameworks prioritize developer experience and make insecure configurations easier than secure ones. Frameworks where the insecure path is harder than the secure path would dramatically improve baseline security across the entire ecosystem. AI agent security gateways, which enforce policy, monitor behavior, and provide observability across agent interactions, should be standardized and made broadly accessible.

(b) In which policy or practice areas is government collaboration with the AI ecosystem most urgent or most likely to lead to improvements in security?

Agent Identity, Authentication, and Authorization Standards (Highest Priority)

This is the single area where government collaboration is both most urgent and most likely to produce transformative results. The inability to establish universal agent identity norms is a classic coordination problem that no individual company can solve but that the government can catalyze. Building on the NCCoE concept paper's approach, the government should convene industry participants around specific technical deliverables: reference architectures for agent identity management; interoperability testing suites; and sector-specific deployment guidance. Federal procurement requirements mandating agent identity standards in acquisitions would create market demand that accelerates private-sector adoption, analogous to the role FedRAMP played in cloud security.

Incident Reporting and Information Sharing

The AI agent security ecosystem currently lacks the information-sharing infrastructure that exists for traditional cybersecurity. The government should establish an AI agent security incident reporting framework, potentially housed within the Cybersecurity and Infrastructure Security Agency and coordinated with existing NIST vulnerability management programs, providing voluntary (and eventually mandatory for critical infrastructure) incident reporting with standardized taxonomies aligned to OWASP and MITRE frameworks, anonymized sharing mechanisms that protect competitive information while enabling pattern recognition, and early warning systems for emerging agent-specific attack patterns.

Critical Infrastructure Protection

As agents are deployed in energy, water, transportation, and healthcare, the potential for cascading failures from compromised or malfunctioning agents demands sector-specific security requirements beyond voluntary guidelines. The CAISI listening sessions planned for April 2026 should produce binding sector-specific security baselines rather than general guidance documents. Defense-focused frameworks being developed under recent National Defense Authorization Act provisions should be coordinated with civilian critical infrastructure requirements to avoid duplicative or contradictory mandates.

Regulatory Coherence and Liability Clarity

Regulatory uncertainty actively harms security investment. When organizations do not know which rules apply, they either over-invest in compliance paperwork or under-invest in actual security. The government should provide authoritative guidance on how existing cybersecurity requirements, including NIST CSF, CMMC, HIPAA Security Rule, and PCI-DSS, apply to agent deployments in their respective sectors. Clear liability allocation for agent actions, specifying which party in the value chain (model developer, framework provider, deployer, user) bears responsibility when agents cause harm, would create appropriate incentives for security investment throughout the ecosystem. International coordination on agent security requirements with the European Union and other major jurisdictions would reduce compliance fragmentation for organizations operating globally.

Security Research Funding

Several critical research areas are pre-competitive and require government investment because their outputs benefit the entire ecosystem and cannot be adequately funded by any individual market participant. Priority areas include: formal methods for verifying agent authorization boundaries; detection methods for multi-agent collusion and coordination failures; security evaluation frameworks for agent supply chains; and privacy-preserving monitoring techniques enabling security observability without violating data protection requirements. Shared testing infrastructure that allows organizations to evaluate their systems against standardized threat scenarios and realistic adversarial attacks without competitive disadvantage would accelerate collective learning significantly.

(c) In which critical areas should research be focused to improve the current state of security practices affecting AI agent systems?

Systematic Threat Modeling and Taxonomies

Before threats can be mitigated, they must be comprehensively characterized. Priority should go to developing empirically grounded taxonomies of attack surfaces unique to agentic AI, distinguishing prompt injection, tool-use hijacking, memory poisoning, goal manipulation, and identity spoofing, and structured threat modeling frameworks analogous to STRIDE and MITRE ATT&CK but tailored to AI agents. Shared vocabulary and attack libraries enable the research and practitioner communities to build on each other's work rather than reconstructing threat models independently.

Adversarial Evaluation at Scale

Controlled, systematic adversarial testing of agent systems in realistic deployment conditions is a critical infrastructure need. This requires automated red-teaming tools capable of probing agent behavior across diverse scenarios, including multi-step attacks that exploit the sequential, stateful nature of agents, with reproducibility, coverage metrics, and open benchmarks the community can iterate on. Evaluation must encompass capability assessment in offensive security domains:

understanding what agents can do when turned against their own systems is necessary for designing appropriate mitigations.

Prompt Injection and Input Integrity

Prompt injection remains the most operationally urgent vulnerability for agents consuming untrusted external content. Research priorities include architectural approaches that separate instruction from data at a fundamental level; content provenance and integrity verification for inputs; and robust detection of injected instructions even when obfuscated, embedded in multimodal content, or distributed across multiple turns of interaction.

Memory and State Security

Agents with persistent memory introduce attack vectors with no analog in traditional software. Poisoning stored context to influence future behavior, exfiltrating information through memory reads, and causing goal drift through incremental manipulation are all under-characterized threats. Research into authenticated and integrity-protected memory systems, anomaly detection over memory evolution, and principled memory access control policies is substantially underdeveloped relative to the risk.

Multi-Agent Threat Dynamics

As ecosystems of interacting agents emerge, research must address how threats propagate across agent boundaries, including cascade failures, adversarial manipulation in agent-to-agent negotiation, and emergent unsafe behaviors arising from agent interactions even when individual agents appear safe in isolation. Simulation environments for studying multi-agent security dynamics at scale would be particularly valuable, as would formal models of inter-agent trust and delegation that can be evaluated for security properties.

Interpretability for Security

Mechanistic interpretability research, developing tools to understand the internal representations and decision processes of agent models, has direct security applications. If it becomes possible to identify when an agent's reasoning has been corrupted, detect the formation of misaligned sub-goals, or trace how a malicious input propagates through an agent's reasoning chain, security monitoring becomes substantially more effective. Prioritizing interpretability research with an explicit security focus would bridge two currently siloed communities and accelerate progress in both.

Robustness, Alignment Under Autonomy, and Sociotechnical Integration

Research is also needed along the following priorities: making agents behave predictably under adversarial conditions and distributional shift; maintaining alignment over extended task horizons where reward modeling faces compounding challenges; and organizational practices for deploying agents responsibly, including incident response frameworks and governance structures that

preserve meaningful human oversight without negating efficiency benefits. The overarching theme is that security and capability should be treated as complementary rather than competing objectives. Research that advances both simultaneously will be most impactful.

(d) How are other countries addressing these challenges and what are the benefits and drawbacks of their approaches?

No country has yet developed a regulatory framework specifically designed for agentic AI systems. Existing approaches vary in structure and emphasis, but all were designed for an earlier generation of AI capabilities.

European Union: Comprehensive Risk-Based Legislation

The EU AI Act classifies AI systems by risk level and imposes escalating compliance obligations, including risk assessment, transparency, human oversight, and robustness requirements for high-risk systems. For agentic AI, the Act provides the strongest legal certainty and clearest accountability chains of any jurisdiction. However, the framework is already encountering implementation strain. The European Commission's Digital Omnibus on AI Regulation Proposal, released in late 2025, noted delays in designating competent authorities and a lack of harmonized standards, and proposed amendments including delayed enforcement and reduced requirements for SMEs. The core tension is pace: the legislation risks becoming outdated as agentic capabilities evolve faster than regulatory text, and compliance costs may disadvantage European firms relative to competitors in less regulated markets.

United Kingdom: Security-Focused Technical Evaluation

The UK's approach is the closest to directly addressing agent security. The AI Security Institute (renamed from the AI Safety Institute in early 2025) has narrowed its focus to security risks including AI-enabled cyberattacks and national security implications, explicitly excluding bias and freedom of speech from its remit. The Institute has tested more than 30 advanced models, stress-tested agentic behavior, developed benchmarks to detect self-replication and sandbagging, and conducted collaborative red-teaming with companies including OpenAI and Anthropic that identified universal jailbreak paths and dozens of safeguard vulnerabilities. Open-source evaluation tools such as Inspect and ControlArena benefit the broader ecosystem. The limitation is enforceability: AISI engagement is voluntary, not all developers participate, and the narrowed security focus leaves broader societal harms unaddressed.

China: State-Directed, Layered Regulatory Control

China has taken a technology-specific approach, enacting targeted regulations for algorithmic recommendation, deep synthesis, and generative AI rather than a single omnibus law. The Cyberspace Administration issued final measures for labeling AI-generated content taking effect in September 2025 and expanded its licensing regime to include foundation-model developers, aligning compliance with data-security and cybersecurity audits. The layered approach allows

rapid regulatory response to emerging capabilities, and China's content provenance requirements directly address manipulation risks that other jurisdictions have been slower to regulate. The approach is difficult for democracies to emulate given its reliance on centralized state authority, and state control over AI development is inseparable from political content moderation objectives.

International Coordination: AI Safety Institute Networks

The International Network for Advanced AI Measurement, Evaluation and Science, coordinated by the UK AI Security Institute in 2026, published its first consensus document on evaluation practices in February 2026 and convened at the India AI Impact Summit to align on next priorities. The network now includes ten jurisdictions. Its value lies in shared evaluation methodologies and joint pre-deployment testing, but effectiveness remains constrained by divergent national priorities. Notably, the UK and US have pivoted toward narrower security concerns, and both refused to sign the Paris AI Action Summit agreement. These developments illustrate how national priorities can fracture coordination in practice. Voluntary participation limits coverage, and geopolitical tensions around Chinese AI development constrain the scope of meaningful cooperation.

Comparative Assessment

The fundamental tension across all approaches is between enabling rapid innovation and ensuring adequate security and resilience, particularly as AI agents become more autonomous. No country has yet developed a regulatory framework specifically designed for agentic AI systems, though the UK's AI Security Institute is closest in terms of direct technical evaluation of agent capabilities and risks. The most effective strategy likely involves combining elements: the EU's legal certainty, the UK's rigorous technical evaluation infrastructure, the U.S.'s innovation dynamism, and the international network's coordination mechanisms, while avoiding the drawbacks of each in isolation.

End of Technical Input

This response represents technical input on the security of AI agent systems for CAISI's consideration. The views expressed reflect practitioner and research perspectives on the current state of the field and priority areas for standards development, research investment, and government-industry collaboration.