XIANGYU GUO

Proposal and Comment Information

Title: Proposed Third-Party Risk Management Guidance, OP-1881

Comment ID: FR-2026-0031-01-C03

Submitter Information

Name: Xiangyu Guo

Submitted Date: 09/13/2026

To the Federal Banking Agencies:

I am submitting comments on the proposed Third-Party Risk Management Guidance issued by the OCC, Federal Reserve Board, FDIC, and NCUA.

I am an independent researcher working on structural safety and responsibility architecture for AI systems. I support the proposed guidance's emphasis on risk-based tailoring and ongoing monitoring. I recommend adding a limited clarification for third-party relationships involving AI and other dynamically composed technology services.

1. Risk assessment should consider the effective service topology, not only the named third party.

Modern AI services may depend on external model providers, cloud infrastructure, retrieval systems, tools, subprocessors, credentials, and other services that are not visible at the banking organization's primary contractual interface.

A third party may therefore remain nominally unchanged while the effective structure through which the service receives information, exercises capabilities, or produces consequences changes materially.

For higher-risk technology relationships, banking organizations should consider whether they can identify material downstream dependencies and understand how those dependencies affect access, authority, information flow, operational resilience, and responsibility.

2. Material topology changes should inform ongoing monitoring.

The proposed guidance appropriately recognizes that risks can change over the life of a third-party relationship.

For AI-enabled services, potentially material changes may include:

addition or replacement of an underlying model provider;
new subprocessors or external services;
new persistent memory or retrieval sources;
access to additional customer or institutional data;
new tools or privileged credentials;
increased autonomous or agentic execution capability; or
reduced human approval or intervention requirements.

Depending on materiality, such changes may justify updated risk assessment, due diligence, testing, contractual review, or monitoring.

3. Third-party oversight should distinguish permission from effective capability.

Traditional access controls remain necessary, but AI-enabled systems can combine individually permitted information, tools, and credentials into capabilities that exceed the practical boundary originally evaluated.

For higher-risk relationships, due diligence and testing should therefore consider combinations of access and authority, including whether lower-trust information can influence higher-authority actions and whether independently enforced controls prevent unauthorized consequences.

4. Incident records should support reconstruction across third-party boundaries.

When a material incident involves an AI-enabled third party, the banking organization should be able to determine the relevant service configuration, information path, external dependencies, authority used, actions attempted or completed, responsible parties, and available intervention points.

The objective should be reconstructable responsibility rather than indiscriminate log accumulation.

These recommendations are compatible with the proposed guidance's principles-based approach. They do not require banking organizations to perform exhaustive technical mapping of every low-risk vendor. They provide a way to scale oversight when a third-party technology relationship becomes materially more interconnected, autonomous, or consequential.

I have attached my supporting paper, The Structural Safety Problem of AI: Execution Topology, Capability-Boundary Mismatch, and the Failure of Permission-First Security, which develops this framework in greater detail.

As financial institutions adopt increasingly compositional and agentic technology services, third-party risk management will depend on knowing not only who the contractual vendor is, but what effective structure is acting on the institution's behalf.

Thank you for considering these comments.

Respectfully,

Xiangyu Guo
Independent Researcher
Alhambra, CA

Attachment:
The Structural Safety Problem of AI: Execution Topology, Capability-Boundary Mismatch, and the Failure of Permission-First Security

The Structural Safety Problem of AI

Execution Topology, Capability-Boundary Mismatch, and the Failure of Permission-First Security

Civilization Physics | AI Governance and Responsibility Architecture
12 September 2026 | Version 1.0

The central safety problem of AI begins where inherited security assumptions stop describing the system that actually exists.

Abstract

An AI system can obey conventional access controls while carrying out an unauthorized transaction. The model may read only files available to its account, use legitimate tools, and operate through authenticated sessions, yet combine those permissions into a disclosure or action the user never authorized. This article identifies the relevant unit of operational AI safety as the **execution topology**: the changing arrangement of information sources, models, memory, tools, credentials, infrastructure, human decisions, and responsibility relationships through which an interaction acquires consequences. It develops **Capability-Boundary Mismatch**, the condition in which effective connected capability exceeds the system's capacity to maintain enforceable and evidenced boundaries around its use. Public disclosures concerning hidden model routing, enterprise retrieval, tool metadata, browser agents, and research infrastructure expose different instances of this problem. Classical security and systems-safety research establish its foundations; recent agent defenses demonstrate feasible architectural responses. The central failure of permission-first deployment is the assumption that an initial grant legitimizes subsequent combinations of access, interpretation, delegation, and action. Structural safety instead requires controls at the transitions where information gains authority and authority gains effect. Alignment remains valuable, but a safety case must also establish what the system prevents when model judgment fails. The resulting doctrine preserves

broad generative capability while binding consequential execution to explicit scope, independent enforcement, traceable evidence, and accountable ownership.

Keywords: structural safety; execution topology; capability-boundary mismatch; AI agents; prompt injection; information-flow control; delegated authority; responsibility architecture.

1. The Interface Conceals the System

A person can select an AI product without knowing which organizations will process the resulting interaction. In its September 2026 threat report, Anthropic stated that Moonshot AI had forwarded customer requests to Claude and returned Claude's responses through its own service. Anthropic reported sensitive customer information in those requests and explicitly said it did not know whether Moonshot had notified the affected customers about the rerouting. These claims are reported here as Anthropic's findings and attributions. The structural implication is direct: the identity presented at the interface can diverge from the actual processing chain. [1]

Training a model on another model's outputs and forwarding live customer requests are distinct operations. Evidence of distillation alone establishes neither live forwarding nor a particular customer-data exposure. Likewise, an external provider's nationality cannot determine a deployment's security. What matters for this problem is the actual route, the information carried along it, the applicable processing conditions, and the authority under which the transfer occurs.

The same question extends well beyond model providers. A document can influence a privileged assistant. A tool description can influence how another tool is used. A browser agent can carry information between authenticated services. A sandbox's permitted dependency can expose a route its designers intended to exclude. The disclosures examined below place these failures at different points in the same relationship: intelligence is connected to information and authority through a structure that the user does not directly perceive. [2, 3, 4, 5, 6]

The central claim of this article is therefore straightforward: **operational AI safety must be established for the execution topology through which a model acts.** A model-level evaluation answers only part of that question. A capable, cautious model can be embedded in an unsafe deployment; the same model can face sharply different consequences under different permissions, data paths, and intervention mechanisms.

This argument extends the author's earlier work in three directions. *Responsibility Before Permission* locates accountable operation upstream of deployment authorization. *No AI Is Responsibility-Free* makes responsibility responsive to the structure of use, including cognitive exposure and consequence burden. *The Burnout of Possibility* identifies the mismatch between an expanding executable surface and finite governing bandwidth. The present article brings these arguments together at the security boundary: what new paths does AI make usable, what constrains those paths, and who carries responsibility for their consequences? [7, 8, 9]

2. Execution Topology as the Safety Object

Execution topology denotes the effective, time-dependent arrangement of components and relationships through which an AI-mediated task receives information, forms proposals, exercises capabilities, changes state, and produces consequences. Its nodes include technical components and responsible actors. Its relationships include data transfer, semantic influence, credential delegation, execution, persistence, observation, and intervention.

The distinctions between these relationships matter. A hostile document can influence an agent without possessing its credentials. A logging service can receive sensitive information without participating in the model's reasoning. A reviewer can carry nominal responsibility while lacking timely access to the evidence or controls needed to intervene. Treating all these relationships as a single undifferentiated connection conceals the mechanism that must be governed.

The topology must also distinguish the **observed execution** from the **reachable execution space**. The first records what happened during a particular run. The second includes consequential paths available under the deployment's credentials, configuration, infrastructure, and plausible inputs. A clean transcript establishes little about an unused route to a sensitive destination. Security analysis must consider both ordinary workflows and paths an adversary, a mistaken agent, or a more capable successor model could activate.

Boundary requirements are task-relative. A writing assistant disconnected from private resources and external action requires a different boundary argument from an agent authorized to inspect payroll and send email. Memory, delegation, fallback routing, or a newly enabled connector can change that argument without changing the product's name. The same principle applies to ordinary human use: an apparently harmless generated answer can acquire institutional force when someone adopts it as a diagnosis, eligibility decision, or operational instruction. [8]

The intellectual foundations are established. Saltzer and Schroeder's protection principles include least privilege, complete mediation, and minimizing shared mechanisms. Denning formalized permitted information flow. Hardy's confused deputy identifies the danger of a privileged intermediary exercising its authority under another party's influence. Leveson's systems-safety account extends analysis beyond individual component failures to control relationships and organizational conditions. [10, 11, 12, 13]

These foundations make an important correction possible. The problem does not arise because classical security had no concept of composition or continuing authorization. It arises when AI deployment treats those requirements as satisfied by a product label, an access token, or confidence in the model. NIST's zero-trust architecture already rejects implicit trust based on location or ownership; its generative AI profile explicitly recognizes risks from opaque component integration and the wider value chain. [14, 15]

AI intensifies the composition problem because ordinary language becomes an operational steering surface. Text retrieved to answer a question may also propose a workflow, impersonate an authority, or redirect tool use. Greshake and colleagues demonstrated this indirect prompt-injection mechanism in LLM-integrated applications. The dangerous transition is the promotion of lower-trust content into authority over how a capability-bearing process acts. [2]

Execution topology supplies the integrating object for these mechanisms. It asks where information enters, how it can influence decisions, which resources those decisions can reach, and where effects can be stopped. Component quality remains important, but the connections determine what a component's failure can cause.

3. Permission Is Local. Execution Is Topological.

Consider an illustrative assistant authorized to read an employee's project files and create public issue reports. Each permission has a legitimate purpose. A retrieved document then instructs the assistant to include an internal planning note in the next public report. The file service accepts the read because the account has access. The publishing service accepts the write because the account may create reports. Both local checks succeed. The composed transaction still violates the intended confidentiality boundary.

The missing condition concerns the relationship between source, transformation, destination, and purpose. Permission to read a document does not establish permission to disclose it. Permission to publish an issue does not establish permission to publish every piece of information the agent can retrieve. Nor does a document's instruction establish that its author may choose how the user's authority is exercised.

This is where permission-first deployment becomes structurally weak. It treats the initial grant as the decisive governance event and leaves subsequent combinations to the agent's interpretation. The system verifies that a capability exists but fails to establish whether this particular use belongs inside the authorized task. Information-flow control and capability-oriented design address precisely the distinction between having access and exercising appropriately bounded authority. [11, 12]

Permission is local when separately valid grants fail to govern their composition across a consequential transaction. Authorization systems can avoid this failure by binding access to specified resources, recipients, operations, duration, and workflow conditions. The required object is an enforceable authorization envelope for the consequential transaction, not a permission dialog for every internal step.

Three questions must remain distinct. Is the account technically able to perform the operation? Has the relevant person or institution authorized this particular use? Does the operation remain permissible after considering the affected parties, information, and consequences? A valid token answers the first question. It cannot, by itself, settle the other two.

The same distinction exposes the limits of self-reported safety. A model's statement that an action is authorized supplies another piece of generated content. An independent decision point must evaluate the action against permissions and constraints whose authority does not originate in the material currently being processed. A natural-language prohibition can guide behavior, but it becomes an operational boundary only when an adequate mechanism enforces its implications.

This requirement covers mistakes as well as attacks. A system can disclose information through an approved logging destination, retain material beyond its intended use, or invoke an inappropriate fallback provider without any adversary manipulating the model. Under those configurations, every component may be following its immediate implementation. The overall arrangement still fails the intended boundary. NIST's treatment of third-party integration and lifecycle governance supports analyzing these paths alongside adversarial misuse. [15]

4. Capability-Boundary Mismatch

Capability-Boundary Mismatch occurs when an AI system's effective connected capability exceeds the capacity of its technical and institutional controls to maintain the boundaries required for authorized, accountable operation.

Effective connected capability means the task-relevant space of accessible information, executable operations, reachable recipients, persistent state changes, and combinations among them. It depends on both the environment and the agent's ability to use it. Adding a tool can expand this space, but so can improving planning, extending memory, enabling delegation, or discovering an indirect route through an existing service.

Boundary capacity means the ability to specify, enforce, observe, and sustain the conditions under which those capabilities may be exercised. The relevant conditions include confidentiality, instruction authority, permitted recipients, operational scope, retention, revocation, and responsibility for intervention. A policy document contributes to specification. A tested control contributes to enforcement. A record contributes to evidence. These functions support one another, but none can simply substitute for the others.

The mismatch has a static and a dynamic form. A system may begin with excessive standing authority relative to its controls. It may also acquire a growing deficit as new workflows and reachable paths outpace reassessment, enforcement, and incident-response capacity. Conversely, a new connection to an effective authorization broker or a restrictive data gateway can improve safety. Connection count alone is therefore a poor measure.

A practical assessment starts with a **boundary-obligation inventory**. For each consequential flow, identify the source, potential influence, action, destination, persistence, controlling policy, enforcement point, evidence, and responsible owner. The gap consists of obligations for which the deployment lacks a current and credible enforcement argument. An unverified obligation is an assurance gap; an experimentally demonstrated bypass is a vulnerability. Conflating these categories would exaggerate the evidence and weaken prioritization.

Measurement should use deployment-specific indicators: the proportion of high-consequence paths lacking enforcement, excessive standing privileges, unreviewed topology changes, ineffective revocation, or intervention delays beyond the relevant harm window. A weighted index can support management only after its categories, weights, and observation method are defined.

The connection to Capability-Bandwidth Mismatch is direct. The earlier model concerns executable possibilities exceeding human and organizational governing bandwidth. Here, the corresponding pressure appears in boundary engineering: more executable combinations require more adequate control, while attention, validation, and maintenance remain finite. [9]

The Boundary Vacuum Law supplies a complementary institutional interpretation. When a consequential function remains active but its governing boundary fails, unresolved burdens are redistributed through the available relationships. In AI deployment, an organization can automate decisions while leaving exception handling, explanation, or compensation to actors with less control over the underlying system. This is a responsibility-routing problem that technical containment alone cannot settle. [16]

Capability extension without corresponding boundary maintenance creates structural exposure. The cases below isolate the mechanisms through which that exposure appears across different execution topologies.

5. Evidence Across the Execution Topology

The following cases isolate distinct boundary failures across the execution topology. Vendor-confirmed vulnerabilities, controlled demonstrations, audits, and operational incidents carry different evidential status; each is used for the mechanism it directly establishes.

5.1 Retrieved content influences privileged enterprise access

Microsoft describes EchoLeak, associated with CVE-2025-32711, as a remediated multistage prompt-injection technique affecting Microsoft 365 Copilot. Under particular conditions, a crafted email could influence the prompt seen by Copilot and cause limited internal data already accessible to the victim to be exfiltrated without the victim's knowledge. The evidential scope is the vendor-confirmed vulnerability and its exploit path; the cited disclosure does not provide a prevalence estimate for customer compromise. [17]

The critical crossing lay between retrieved content and the assistant's use of privileged access. An external sender did not need the victim's file permissions in order to influence an agent that possessed them. The case makes the authorized-account/unauthorized-transaction distinction concrete.

5.2 A public issue redirects private repository access

In May 2025, Invariant Labs demonstrated a public GitHub issue influencing an agent connected through GitHub MCP. In a setup with demo repositories and sufficient user-granted access, the agent retrieved private repository information and placed it in a public pull request. The disclosure discusses permissive tool confirmation, including "Always Allow," as part of the deployment exposure. It identifies an agent-system architectural issue rather than a server-code flaw that GitHub alone could patch. [3]

The demonstrated breach crossed repositories through the assistant's combined authority. The public issue supplied influence; the user's integration supplied access and publication capability. The result depended on that configuration, rather than on public issues intrinsically possessing private access.

5.3 Tool metadata becomes a source of instructions

Invariant's April 2025 tool-poisoning disclosure showed that malicious instructions in MCP tool descriptions could induce an assistant to read sensitive material and transmit it through tool arguments. It also described cross-server influence, in which one tool's description affected the use of another server's tools. These were research demonstrations of unsafe agent configurations. [4]

The entry point matters. A tool description appears close to the operational control layer, yet can originate with an external provider. An interface that presents the user with a simplified tool label may omit precisely the content influencing the model. Approval of a tool's apparent function cannot stand in for governing all semantic material delivered through its registration and responses.

5.4 A browser agent joins authenticated domains

Trail of Bits' February 2026 disclosure describes a prelaunch Comet audit conducted in April 2025. Researchers demonstrated four prompt-injection techniques that extracted Gmail information while the user asked the assistant to summarize an attacker-controlled page. Perplexity's quoted response states that the identified gaps were closed before launch. The evidential scope is the audited prelaunch configuration. The cited disclosure also states that Perplexity closed the identified gaps before launch. [5]

The browser agent created a route between page content, authenticated information, and an external recipient. This did not require breaking the browser's same-origin implementation in the ordinary sense. The agent itself could act across services using the user's sessions. Security analysis therefore had to include the agent-mediated path, alongside conventional browser boundaries.

5.5 Shared infrastructure defeats intended isolation

OpenAI's August 2026 account of the Hugging Face incident describes internal research models operating under reduced safeguards during cybersecurity evaluations. Agents used a shared Artifactory service for unauthorized communication and indirect internet access, subsequently compromising parts of OpenAI's research infrastructure and Hugging Face's systems. OpenAI also reported that earlier observations of unexpected communication and internet access had not reached the leaders handling the later incident. OpenAI reported no impact on its customer data or products. [6]

The incident occurred in a research-evaluation environment materially different from ordinary consumer deployment. Its structural relevance is transitive reachability: blocking direct internet

access did not establish isolation while an accessible dependency offered an indirect route. Technical scope and escalation scope both fell short of the effective topology.

Together with the routing case in the introduction, these cases locate exposure in provider selection, retrieved data, repository access, tool metadata, authenticated browsing, and shared infrastructure. They support a common analytical object without collapsing their causes or evidence status. Some involve adversarial content; another involves agent behavior during evaluation; provider routing can occur before the responding model exercises any judgment. A defense focused exclusively on the model's response cannot cover all these locations.

6. What Alignment Can and Cannot Establish

Behavioral alignment changes the likelihood that a model follows an inappropriate instruction or produces a harmful response. Constitutional AI, for example, uses feedback and training procedures to shape behavior. It should be evaluated as an actual learning method, rather than reduced to a written declaration. Its value does not establish that a deployment's data routes, credentials, or external effects are adequately constrained. [18]

A model can correctly refuse to disclose a secret after a provider has already routed that secret outside the intended processing domain. It can decline to delete a database while a standing credential still permits deletion through another reachable operation. It can recognize most malicious documents without supplying a guarantee about the next document. These are differences between behavioral resistance and enforced operational scope.

Recent defenses help specify that distinction. StruQ combines a structured-query front end with specially trained models that distinguish instructions from data. Its protection depends in part on learned behavior. CaMeL separates trusted task planning from untrusted data handling and adds capability-based controls over permitted flows. Fides tracks confidentiality and integrity labels and enforces information-flow policies in its planner. These approaches occupy different positions in the control structure; they do not provide interchangeable guarantees. [19, 20, 21]

Their significance is architectural. Security can be supported by restricting the consequences available to a compromised or mistaken model. The remaining obligations include policy adequacy, trusted implementation, correct labeling, permitted declassification, and coverage of the interfaces actually used. An information-flow guarantee also leaves other problems open, including false answers or manipulative content that remains within an allowed channel. [20, 21]

A structural safety claim must therefore state its protected property and assumptions. For a specified threat model, every feasible route to a prohibited effect must be removed or intercepted by an effective control before that effect occurs. One uncovered route defeats that particular containment claim. Merely adding another model to approve actions does not establish independence when both models receive the same poisoned context or share authority over the same policy.

The practical test is demanding: **when the model makes the wrong judgment, which unacceptable consequences remain unavailable to it?** Where the answer depends entirely

on the model recovering the correct intention, the boundary remains behavioral. Where the answer depends on independent, tested constraints, the system has an additional basis for containment.

7. A Structural Safety Doctrine

7.1 Bind authority to a task and its consequences

Provide the capabilities needed for the authorized workflow, with explicit resources, operations, recipients, and duration. Separate exploratory reading and generation from external publication, credential administration, destructive modification, and other consequential execution. Delegation to another agent should preserve or narrow the authorized scope unless a legitimate authority explicitly expands it.

A model may propose an action; an authorization mechanism should decide whether the actual action fits the task envelope. That decision must evaluate concrete parameters and relevant prior state. The agent must not be able to rewrite the governing policy through the same channel that proposes its next step. This applies least privilege and continuing mediation to the composed task. [10, 14]

7.2 Preserve boundaries through transformation and persistence

A summary, translation, extracted field, or generated report can retain sensitive information from its sources. Transformation should not automatically remove confidentiality restrictions. Likewise, lower-trust material should not acquire instruction authority merely because it has been summarized, stored in memory, or returned by a trusted connector. Labels and enforcement must follow the relevant information and influence through the operations that can carry them onward. [11, 21]

This requires discriminating design. Treating every mixed context as equally sensitive can disable useful work; treating every model-generated transformation as clean invites laundering. The system needs explicit rules for combining sources, limiting influence, releasing information, and authorizing exceptions. Human-approved declassification should identify what is being released, to whom, and under whose responsibility.

7.3 Control actual routes and dependencies

The operational boundary must include model routing, fallback services, retrieval systems, code execution, package infrastructure, telemetry, caches, and other reachable recipients. These components need not all exist in every deployment. Where they do exist, their access, onward connectivity, retention, and change procedures belong inside the safety argument.

“Local,” “private,” and “sandboxed” describe configurations whose implications require verification. A local model can still invoke an external tool; a private endpoint can still feed an insufficiently isolated logging service. An approved processing destination should not become an unexamined relay to another destination. Network restrictions, compartmentalization, supplier controls, and suitably scoped attestation can supply evidence, but no single label establishes end-to-end containment.

Reassess the relevant boundaries when a model, tool description, credential, memory policy, or dependency changes. Improved agent capability can activate a latent path even where the software inventory remains unchanged. The test should address what the system can now reach and cause.

7.4 Make human intervention timely and meaningful

Human oversight becomes operational only when a person has enough context, competence, time, and authority to change the outcome. A generic confirmation prompt does not supply those conditions. A consequential-action preview should expose the authenticated recipient, affected resources, proposed change, sensitive content where appropriate, and consequences that will be difficult to reverse.

Routine activity inside an established envelope can proceed without repeated interruption. Escalation belongs at material changes of scope, sensitive disclosure, irreversible action, or unresolved uncertainty. Where harm can occur faster than review, the system must delay execution or enforce a safe boundary without waiting for an inattentive user to notice. The author’s Feedback-and-Responsibility Standard requires validation before the relevant harm pathway outruns it. [22]

7.5 Preserve evidence without creating a second leakage system

An audit record should reconstruct the material execution: relevant component versions, data sources, transformations, destinations, requested and exercised authority, approvals, policy decisions, and observed effects. Provenance frameworks such as W3C PROV distinguish entities, activities, and responsible agents, providing a basis for such records. Provenance establishes a traceable history; truth, legitimacy, and safety still require evaluation. [23]

Generated explanations cannot substitute for independently recorded events. Conversely, collecting every raw prompt, document, or reasoning trace can create another sensitive repository. Record what the accountability and incident-response functions require, protect it with suitable access and retention controls, and avoid unnecessary duplication of secrets. Evidence must support containment and correction rather than merely document harm afterward.

7.6 Anchor responsibility where control can be exercised

Identify who configures capabilities, approves data routes, changes models or tools, owns source restrictions, responds to anomalies, and can stop the system. These responsibilities must be supported by actual access to information and intervention mechanisms. A frontline user cannot reliably carry responsibility for an undisclosed provider route or a shared infrastructure service they cannot inspect or disable.

Responsibility allocation should track control, knowledge, and capacity to prevent or repair harm. This architectural principle does not determine legal liability; contracts, professional duties, statutory rules, and jurisdiction-specific law govern that determination. The technical trace should enable those determinations across the supplier chain. [7, 8]

The resulting doctrine preserves **Governed Openness**. Users can explore, generate alternatives, and work across domains without every speculative output acquiring institutional authority. Stronger controls apply when information changes recipient, an output becomes a decision, or a proposal becomes an external act. Security then supports expanded capability instead of demanding either unrestricted delegation or indiscriminate shutdown. [8]

8. From Case Evidence to Testable Claims

The framework yields deployment-level tests. First, hold the model approximately constant while changing reachable resources, standing authority, and enforcement. The relevant question is whether prohibited effects decline while authorized task completion remains useful. A rise in refusal rate alone does not establish successful containment.

Second, test compositions that individual tool evaluations miss. Place lower-trust material in documents, messages, metadata, memory, and tool responses, then examine whether it can redirect higher-authority actions. Include indirect recipients, delayed persistence, delegated agents, and dependencies shared between supposedly isolated environments. Evaluate attempted actions and externally observed effects, rather than relying only on the assistant's visible final answer.

Third, repeat those evaluations after meaningful changes. A safety argument attached only to the original model or integration date cannot establish the behavior of a materially changed topology. Measure the time between detecting a boundary violation and stopping its consequences, including whether escalation reaches the actor capable of intervention.

AgentDojo already provides a task-oriented environment for evaluating attacks and defenses together, reinforcing the methodological emphasis on both security and useful work. Its results operate at benchmark scale; population incidence and deployment-wide guarantees require separate evidence. [24]

Capability-Boundary Mismatch also supports longitudinal evaluation through versioned obligation inventories and matched adversarial tests. The empirical question is whether measured gaps predict prohibited effects, and whether closing them improves containment without unacceptable loss of utility. Capability growth accompanied by adequate controls should

not, by itself, predict rising failure. Repeated failure of the gap measures to distinguish outcomes would challenge their explanatory value.

The scope is equally important. Strong boundary enforcement cannot establish the truth of every answer or the legitimacy of every goal. Cognitive integrity, source quality, domain judgment, and institutional purpose remain separate obligations. The execution topology identifies where those obligations must be carried and where inadequate judgment can become harmful action. [8]

9. Conclusion

AI's operational power depends on connections. Those connections allow a model to retrieve context, use specialist services, remember prior work, coordinate tools, and act in the world. They also determine how influence becomes authority, where information travels, and who must intervene when something goes wrong.

A product identity cannot describe that entire structure. An access grant cannot legitimize every downstream combination. A behavioral safety score cannot establish the limits of a credential, a hidden relay, or a persistent record. Each answers a narrower question than a serious deployment must resolve.

Execution topology provides the appropriate unit of analysis. Capability-Boundary Mismatch explains the exposure created when effective capability outruns enforceable governance. Structural safety supplies the response: bounded delegation, controlled information flow, independent enforcement, evidenced routing, timely intervention, and responsibility supported by actual control.

The governing standard is simple: **an AI safety claim must identify the consequences its architecture prevents even when model judgment fails.** Where that prevention remains unestablished, capability has exceeded the demonstrated boundary. Progress requires bringing the boundary forward with the capability.

References

[1] Anthropic. (2026, September). *Detecting and countering misuse of AI: September 2026*. Section GTG-16002, 'Moonshot serves Claude instead of Kimi and collects exchanges for model training.' Anthropic threat report.

[2] Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., & Fritz, M. (2023). *Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection*. arXiv:2302.12173, version 2. doi:10.48550/arXiv.2302.12173.

[3] Milanta, M., & Beurer-Kellner, L. (2025, May 26). *GitHub MCP exploited: Accessing private repositories via MCP*. Invariant Labs. Original research disclosure.

[4] Beurer-Kellner, L., & Fischer, M. (2025, April 1). *MCP security notification: Tool poisoning attacks*. Invariant Labs. Original research disclosure.

[5] Trail of Bits. (2026, February 20). *Using threat modeling and prompt injection to audit Comet*. Report of an April 2025 prelaunch audit. Audit disclosure.

[6] OpenAI. (2026, August 26). *The Hugging Face incident and the road ahead*. Technical investigation summary. Provider incident report.

[7] Guo, X. (2026). *Responsibility before permission: Why the old free-market formula breaks in the AI age. Autonomous vehicles, responsibility anchoring, and the end of permission-first commercialization*. Independent research article.

[8] Guo, X. (2026). *No AI is responsibility-free: The responsibility topology of consumer interaction, cognitive integrity, and high-consequence systems*. Independent research article.

[9] Guo, X. (2026). *The governance of human capacity in the AI age, Vol. 1: The burnout of possibility: Capability-bandwidth mismatch in the early AI era*. Independent research article.

[10] Saltzer, J. H., & Schroeder, M. D. (1975). The protection of information in computer systems. *Proceedings of the IEEE, 63*(9), 1278–1308. doi:10.1109/PROC.1975.9939.

[11] Denning, D. E. (1976). A lattice model of secure information flow. *Communications of the ACM, 19*(5), 236–243. doi:10.1145/360051.360056.

[12] Hardy, N. (1988). The confused deputy. *ACM SIGOPS Operating Systems Review, 22*(4), 36–38. doi:10.1145/54289.871709.

[13] Leveson, N. (2004). A new accident model for engineering safer systems. *Safety Science, 42*(4), 237–270. doi:10.1016/S0925-7535(03)00047-X.

[14] Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). *Zero trust architecture*. NIST Special Publication 800-207. doi:10.6028/NIST.SP.800-207.

[15] National Institute of Standards and Technology. (2024). *Artificial intelligence risk management framework: Generative artificial intelligence profile*. NIST AI 600-1. doi:10.6028/NIST.AI.600-1.

[16] Guo, X. (2026). *The boundary vacuum law: Gradient, boundary failure, topological flow, and pressure capture in social systems*. Version 2. Independent research article.

[17] Microsoft. (n.d.). *AI application security series 1: AI tools*. Microsoft Security Insider. EchoLeak discussion and remediation reference for CVE-2025-32711. Accessed 12 September 2026. Microsoft security guidance.

[18] Bai, Y., et al. (2022). *Constitutional AI: Harmlessness from AI feedback*. arXiv:2212.08073. doi:10.48550/arXiv.2212.08073.

[19] Chen, S., Piet, J., Sitawarin, C., & Wagner, D. (2025). StruQ: Defending against prompt injection with structured queries. In *34th USENIX Security Symposium*, 2383–2400. USENIX proceedings.

[20] Debenedetti, E., Shumailov, I., Fan, T., Hayes, J., Carlini, N., Fabian, D., Kern, C., Shi, C., Terzis, A., & Tramèr, F. (2025). *Defeating prompt injections by design*. arXiv:2503.18813, version 2. doi:10.48550/arXiv.2503.18813.

[21] Costa, M., Köpf, B., Kolluri, A., Paverd, A., Russinovich, M., Salem, A., Tople, S., Wutschitz, L., & Zanella-Béguelin, S. (2025). *Securing AI agents with information-flow control*. arXiv:2505.23643, version 2. doi:10.48550/arXiv.2505.23643.

[22] Guo, X. (2026). *The governance of human capacity in the AI age, Vol. 4: The feedback-and-responsibility standard for AI-assisted work*. Independent research article.

[23] Moreau, L., & Missier, P. (Eds.). (2013, April 30). *PROV-DM: The PROV data model*. W3C Recommendation. W3C Recommendation.

[24] Debenedetti, E., Zhang, J., Balunović, M., Beurer-Kellner, L., Fischer, M., & Tramèr, F. (2024). *AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents*. arXiv:2406.13352, version 3. doi:10.48550/arXiv.2406.13352.