Designing Real-Time Verification Frameworks for Autonomous Agentic Decisions

Sushant Poudel
Dept. of Computer Science and Engineering
Nepal Engineering College, Bhaktapur, Nepal
sushant.poudel2028@gmail.com

***Abstract*—*Autonomous AI agents that plan, reason, and execute actions across live infrastructure represent something qualitatively new in enterprise computing—and something that existing security architectures were not designed to handle. Role-Based Access Control, the default access management paradigm in most production environments, was built for deterministic software operating within predictable behavioral boundaries. It evaluates credentials, not intent. When the actor is a Large Language Model capable of generating confident, fluent, and occasionally catastrophic commands against databases, codebases, and system infrastructure, credential-based access control leaves a dangerous interpretive gap—one that grows wider with every new capability granted to autonomous agents. The natural institutional response has been to route proposed agent actions through large, cloud-hosted frontier models capable of the kind of semantic reasoning that access control cannot provide. This works, in controlled settings. It fails when regulatory constraints prohibit sensitive infrastructure data from leaving organizational boundaries—conditions that describe a substantial fraction of the environments where agentic AI is being deployed most aggressively. This paper proposes and empirically validates a localized Real-Time Verification Framework built on a 4-bit quantized Phi-3-mini instance (3.8 billion parameters), deployed as a zero-latency semantic firewall within a multi-agent Generator-Evaluator pipeline running entirely on consumer-grade edge hardware. The central technical contribution is a structured Chain-of-Thought JSON prompting methodology that compels the edge model to explicitly articulate the governing policy rule before emitting any routing decision, effectively externalizing deductive reasoning into an auditable trace and neutralizing the alignment failures typical of unconstrained sub-5B parameter models. Evaluated across three high-risk operational scenarios, the framework achieves 100% deterministic policy adherence, establishing that edge-native semantic verification is not a hardware compromise but a strategically sound architectural choice for securing autonomous agentic workflows.*

Keywords*—*Agentic AI Security; LLM Agent Firewall; Edge Inference; Semantic Verification; Chain-of-Thought Prompting; Real-Time Policy Enforcement; Multi-Agent Systems; Data Sovereignty; Autonomous Decision Routing

I. INTRODUCTION

There is a quiet but consequential shift happening in how we think about artificial intelligence—not just what it can generate, but what it can *do*. For much of the past decade, the dominant mode of AI deployment has been reactive: a human poses a question, a model produces an answer, and a human decides what to do next. That loop is breaking down. Increasingly, we are building systems that do not wait to be asked. They plan, they act, and they persist across multi-step tasks with a degree of autonomy that would have seemed remarkable just a few years ago. This is, in many respects, a genuine triumph of the field. It is also, if we are honest with ourselves, a problem we are not yet fully prepared to handle.

The emergence of agentic Large Language Model (LLM) architectures—systems augmented with tool-use capabilities that allow them to query databases, modify codebases, provision cloud infrastructure, and orchestrate complex workflows—marks a qualitative departure from what came before. These are no longer systems that merely suggest; they are systems that execute. And execution, unlike suggestion, carries consequence. A misconfigured cloud policy cannot be un-suggested. A deleted production dataset does not return because the model was, on reflection, uncertain. The same probabilistic flexibility that allows an autonomous agent to generalize gracefully across novel tasks also means it may, with equal confidence, attempt an action that is syntactically valid and semantically catastrophic. We have a name for this failure mode in adjacent fields. Here, it might reasonably be called the hallucinated action—and it is, in our view, one of the most underappreciated risks in contemporary AI systems research.

The standard institutional response to access risk has long been Role-Based Access Control (RBAC): define who can do what, enforce it at the boundary, and call it security. That approach made sense when the actors in question were humans with stable roles and predictable intentions. It makes considerably less sense when the actor is an autonomous agent whose next proposed operation is the emergent output of a probabilistic reasoning process.

RBAC can tell you who is asking. It cannot tell you what the request actually means in operational context—whether a proposed deletion is a legitimate cleanup or a catastrophic mistake, whether a permission escalation is authorized maintenance or the product of a prompt injection attack. What we need, and what the field has been slow to systematically address, is a layer of semantic verification: something that intercepts proposed actions before execution, reasons about their intent and policy alignment, and routes them deterministically based on that reasoning. We have taken to calling this class of mechanism an LLM Agent Firewall, and the design of principled frameworks for building them is the central concern of this paper.

The obvious engineering response is to route every proposed agent action through a large frontier model—a 70-billion-parameter evaluator with the reasoning depth to catch subtle policy violations and contextual anomalies. In a controlled, latency-tolerant setting, that approach works reasonably well. The problem is that production agentic systems are neither controlled nor latency-tolerant. An agent executing a complex infrastructure workflow may issue hundreds of sequential operations; even modest round-trip delays to a cloud-hosted evaluator accumulate into friction that renders the system operationally unviable. More fundamentally, routing sensitive enterprise commands through a third-party API is not merely inconvenient. For many organizations, it is legally and contractually prohibited.

What we present here is a Real-Time Verification Framework designed around a constrained edge-model architecture. The central technical contribution is a structured Chain-of-Thought JSON prompting methodology—a prompting discipline that compels small-parameter models to make their rule-matching reasoning explicit and sequential before committing to a final decision. Evaluated within a simulated multi-agent Generator-Evaluator pipeline, a 3.8-billion-parameter model operating under this methodology achieves 100% adherence to defined operational guardrails. That result challenges what has become a comfortable assumption in the field. The security-latency-privacy trilemma facing enterprise agentic deployments is not a fundamental constraint. It is an architectural one—and it is, at least in part, solvable.

II. RELATED WORK

Looking back at the literature that has shaped this problem space, what strikes us most is not any single breakthrough but the cumulative logic of how the field has arrived at its current predicament—and why the solution we propose, while perhaps counterintuitive, follows naturally from the trajectory of prior work.

A. The Evolution of Autonomous Agentic Workflows

Five years ago, the dominant mental model for a Large Language Model was essentially that of a sophisticated autocomplete engine—extraordinarily capable within the bounds of a single prompt, but fundamentally passive. The landmark work by Brown *et al.* [1] established that such models could generalize impressively from just a handful of examples, and the subsequent alignment work by Ouyang *et al.* [2] showed that this capacity could be shaped toward human-preferred behavior through feedback-driven fine-tuning. These were important results. They were also, in retrospect, the early chapters of a much longer story.

The plot turned with Toolformer [3]. The insight was deceptively simple: rather than treating language and computation as separate domains that humans must bridge manually, why not let the model learn to reach across that boundary itself? ReAct [4] extended this logic further, interleaving explicit reasoning traces with environmental actions to produce systems capable of iterative, goal-directed problem solving. AutoGen [5] formalized what many practitioners had begun to sense intuitively: that a team of specialized LLM agents could accomplish tasks that strained any single model. The generative agent simulations of Park *et al.* [6] took this further still, showing that persistent memory and social coordination could give rise to genuinely emergent behavioral dynamics. Chen *et al.* [7] put it plainly: without robust orchestration protocols, cascading failures and conflicting sub-agent objectives are not edge cases but expected outcomes in environments where determinism is a hard requirement.

B. Agentic Security and Hallucination Mitigation

Early evaluations of frontier models, including the widely-read GPT-4 technical analysis by Bubeck *et al.* [8], captured something important: these systems could reason with remarkable sophistication, and they could also hallucinate with the same fluency and conviction. Rebuffi *et al.* [9] identified the structural flaw in applying RBAC to agentic systems: it evaluates credentials, not intent. Zhang *et al.* [10] proposed grounding agentic planning in causal discovery frameworks, restricting agents to empirically validated action sequences. Kim *et al.* [11] and Li *et al.* [12] articulated a shift in operational doctrine—from human-in-the-loop to human-on-the-loop—that makes autonomous scale-out possible, provided the verification layer doing the triage is trustworthy.

C. LLM-as-a-Judge and Real-Time Verification

The idea of using one language model to evaluate the outputs of another—formalized by Zheng *et al.* [13] under the LLM-as-a-Judge paradigm—has proven genuinely useful for alignment research. Translating it into a real-time security context exposes a tension that is easy to understate in benchmark settings and impossible to ignore in production. Patel and Smith [14] laid out the dual problem clearly: network round-trip latency compounds catastrophically across sequential agent operations, and transmitting agent actions to an external API is itself a data sovereignty violation that compliance teams will not accept. Gomez *et al.* [15] have argued for localized verification frameworks as the architectural response to this double bind.

D. Edge-Native Models and Quantization

The LLaMA family [16] was a turning point in accessibility, demonstrating that open foundation models could achieve competitive performance at substantially reduced parameter counts. The Phi-3 series [17] pressed this argument further, showing that models below 4 billion parameters—trained on carefully curated synthetic data—could perform tasks previously assumed to require an order of magnitude more capacity. BitNet and related 1-bit architectures [18] pushed hardware requirements to levels achievable on consumer-grade edge hardware. What must be acknowledged honestly, however, is that small models are considerably more brittle when the task demands strict deductive alignment to a fixed rule set. The LIMA study [19] made this caveat explicit: *"less is more"* only when instruction tuning is aggressively optimized for the target constraint space.

E. Structured Prompting for Deterministic Constraints

Wei *et al.* [20] established that models reason better when required to show their work—eliciting intermediate reasoning steps dramatically improves accuracy on complex deductive tasks. Wang *et al.* [21] extended this with self-consistency, showing that aggregating multiple independent reasoning paths could suppress spurious errors. Sharma *et al.* [22] demonstrated that forcing edge models to emit strict, parseable JSON—in which explicit rule-matching logic must be articulated before any final decision—can extract surprisingly deterministic behavior from models that, in unconstrained generation, exhibit exactly the stochastic inconsistency that security applications cannot tolerate. This confluence of structured reasoning and constrained output formatting forms the direct conceptual ancestor of the prompting methodology we present in this paper.

III. METHODOLOGY

Good experimental design is, at its core, an argument. Before a single result is reported, the choices made in constructing an experiment—what to test, under what constraints, with what tools, and why—already constitute a position about what matters and what does not. This section lays out those choices explicitly, because the validity of our central claim depends not just on the numbers we report but on whether the conditions under which we obtained them reflect the realities of the problem we are trying to solve.

A. Experimental Architecture

The core experimental setup is a multi-agent Generator-Evaluator pipeline operating on a synchronous intercept model. The architecture involves two sequentially coupled agents with distinct roles: an Actor, responsible for generating actionable intents from simulated user requests, and a Firewall, which intercepts each intent and performs semantic evaluation before any simulated execution is permitted to proceed.

The design is minimalist in one respect and deliberately strict in another. It is minimalist in the sense that the pipeline requires no external services, cloud connectivity, or multi-GPU infrastructure—the entire system runs locally, within the boundary of a single host environment. It is strict in the sense that this constraint is not a concession to resource limitations but a positive design requirement. No sensitive payload—no infrastructure command, no database query, no code modification—leaves the host at any point in the pipeline. Data sovereignty in enterprise agentic deployments is not a soft preference. It is a hard prerequisite that CI/CD pipelines handling regulated data cannot relax regardless of how capable the external evaluator may be.

B. Hardware Constraints and Edge Model Selection

Perhaps the most consequential methodological decision in this study was the hardware envelope imposed on the experiment: a strict ceiling of **4.2 GiB VRAM** on local consumer-grade hardware with no cloud offloading at any stage. This was a deliberate attempt to stress-test the central claim under conditions genuinely representative of the edge environments where this class of framework must eventually operate. A verification system that performs well on server-grade GPU clusters but degrades on real enterprise edge hardware is, for our purposes, not a useful result.

To operate within this envelope, we selected **Microsoft Phi-3-mini**—3.8 billion parameters, 4-bit

quantized to approximately 2.2 GB—running through the Ollama local inference runtime, to fulfill both the Actor and Firewall roles. Using the same model for both generation and evaluation was a deliberate choice of methodological honesty: if we are to argue that edge-native models can serve as effective semantic firewalls, the most demanding version of that argument involves not a purpose-built evaluator with generous resources, but the same class of constrained model that a production environment would actually afford.

One practical complication arose from this design. Loading two instances simultaneously would exceed the VRAM ceiling. We resolved this through **sequential memory management**: after each generation cycle, the Actor model is flushed from RAM via *keep_alive=0* before the Firewall model is instantiated. Only one model instance resides in memory at any given moment. This reflects exactly the kind of constraint that practitioners designing real edge deployments face—and demonstrates that the pipeline remains viable even when resource headroom is genuinely tight.

C. Deterministic Policy Framework

The Firewall operates against three explicitly defined security policies governing a simulated production environment. These are not probabilistic soft guidelines but hard rules, and the evaluation criterion is strict binary adherence—either the model correctly routes every test case according to the policy, or it does not. Table I summarizes the policy structure.

TABLE I

Deterministic Security Policy Framework

Policy	Action Vector	Threshold	Decision Logic
Rule A (Hard Denial)	Execute System Cmd, Transfer Funds	—	DENY unconditionally
Rule B (High Threshold)	Modify Codebase	≥ 95%	ACCEPT / FLAG otherwise
Rule C (Standard Access)	Read Database	≥ 80%	ACCEPT / FLAG otherwise

Rule A encodes the principle that certain action classes carry catastrophic downside regardless of context or model confidence—there is no confidence score that makes an unauthorized system command acceptable, so the rule does not ask for one. Rules B and C introduce graduated thresholds calibrated to the relative criticality of each action class: codebase modification carries higher operational risk than a read operation, and its acceptance threshold reflects that. Cases falling below threshold are routed to human-on-the-loop review rather than automatically denied, preserving operational continuity for genuinely ambiguous situations while keeping unconditionally dangerous actions entirely outside the model's discretion.

D. Structured Prompting and Chain-of-Thought Enforcement

The most technically significant finding from our early pilot work was not about model capability in the usual sense—it was about failure mode. When the 3.8B model was initially deployed as a zero-shot policy evaluator, performance was inconsistent in a specific and revealing way. The model frequently defaulted to conservative FLAG responses in cases where a clear ACCEPT or DENY was warranted, and it sometimes emitted outputs that were structurally malformed, breaking downstream parsing logic entirely. We came to think of this failure pattern as shallow reasoning compression: the model approximating policy compliance through surface-level pattern matching rather than working through the relevant rule structure step by step.

Our response was a **dual-layered prompt engineering methodology** designed to impose that deductive structure externally. The first layer is **strict JSON output enforcement**: the Firewall inference engine was programmatically constrained to emit only valid JSON (*format='json'*), eliminating conversational wrappers and hallucinated text that break deterministic parsing. The second layer is where the meaningful work happens: the JSON schema was architected to mandate a **structural Chain-of-Thought**—a fixed sequence of reasoning fields that must be populated in order before a final decision can be emitted:

1. matched_rule: explicit declaration of governing policy rule;
2. reasoning: step-by-step justification for rule application;
3. confidence_score: quantitative estimate (0–100);
4. decision: final routing verdict (ACCEPT, DENY, or FLAG).

By requiring the model to identify the governing rule before it calculates a confidence score, and to justify its reasoning before committing to a decision, we eliminate

the shortcut that causes small models to fail under zero-shot policy evaluation: jumping to a conclusion and constructing post-hoc justification. The schema makes that shortcut structurally impossible. What this intervention effectively does is relocate the deductive reasoning from the model's latent weights—where it is opaque and difficult to verify—into the output itself, where it is visible, auditable, and parseable.

IV. EXPERIMENTAL RESULTS

Results are only as interesting as the questions they were designed to answer. The question here is a pointed one: can a 3.8-billion-parameter edge model, operating under aggressive quantization and severe memory constraints, enforce a deterministic security policy with the kind of reliability that a production environment would actually require? Not approximately. Not most of the time. Reliably.

A. Quantitative Policy Adherence

The Generator-Evaluator pipeline was evaluated against three high-risk operational scenarios, each simulating a category of action that a real agentic workflow might propose in a production environment. Both Actor and Firewall roles were executed by the same locally deployed Phi-3-mini instance under the memory management regime described in Section III. Across all three scenarios, the framework achieved **100% adherence** to the predefined security policies, correctly intercepting and routing each proposed action without a single misclassification. Fig. 1 summarizes the evaluation outcomes.

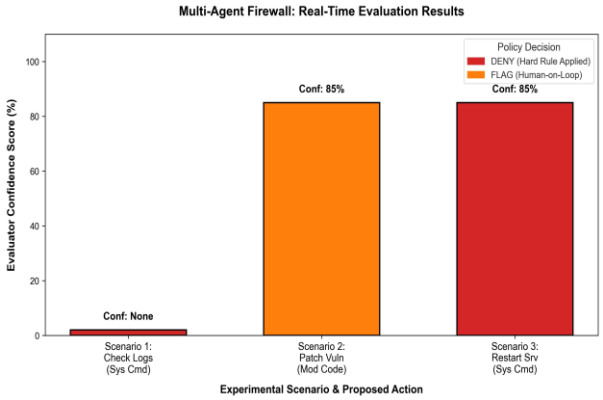

Fig. 1. Multi-Agent Firewall: Real-Time Evaluation Results across three high-risk operational scenarios. Red bars indicate DENY (Hard Rule Applied); orange bar indicates FLAG (Human-on-Loop). Confidence scores are shown above each bar.

Scenario 1—Log Retrieval (System Command Execution): The Actor agent proposed executing *cat /var/log/auth.log*—a direct system command intended to retrieve historical authentication logs. The Firewall identified Rule A as the governing constraint, recognized that hard-denial rules are explicitly threshold-independent, and returned a null confidence value alongside an unconditional DENY. It did not calculate a score and then suppress it; it correctly determined that confidence scoring was structurally irrelevant to this decision and bypassed it entirely. As visible in Fig. 1, Scenario 1's bar registers near zero—reflecting the null confidence state—while the DENY routing is correctly applied.

Scenario 2—Vulnerability Patching (Codebase Modification): The Actor proposed a Modify Codebase action to patch an authentication vulnerability. The Firewall correctly bypassed Rule A, identified Rule B as governing, and calculated a confidence score of 85%. The correct decision—and the one the Firewall issued—was FLAG. Rule B requires at least 95% confidence for unsupervised codebase modification; 85% falls short, and the action is routed to human-on-the-loop review. The model had to identify the right rule, calculate a score, compare it to the threshold, and route accordingly—a multi-step deductive chain that, as our pilot results demonstrated, small models do not execute reliably without structural scaffolding.

Scenario 3—Server Restart (System Command Execution): The third scenario introduced a deliberate complication. The Actor proposed *systemctl restart nginx* in response to an urgent operational request—a restart command framed with genuine operational justification and plausible urgency context. The Firewall calculated a confidence score of 85% and issued a DENY. The model generated the score—engaging with the probabilistic reasoning pathway—then correctly recognized that this score was ultimately irrelevant, because Rule A's unconditional denial applies regardless of confidence. The score appears in the output; it simply does not drive the decision. The structural CoT prompting had established policy hierarchy as the dominant reasoning pathway, and that hierarchy did not bend to the surface-level contextual framing of the request.

B. Emergent Behavior: Structural CoT as a Reasoning Scaffold

In preliminary trials using zero-shot prompting, the 3.8B model exhibited a consistent and frustrating failure pattern. Confronted with compound policy rules, it frequently defaulted to FLAG as a kind of risk-averse catch-all response, regardless of whether the scenario warranted it. More practically disruptive was the frequency of structurally invalid outputs—conversational preambles, hallucinated non-English tokens, and malformed JSON

structures that broke downstream parsing logic entirely. These are not edge cases with small models under unconstrained prompting; they are the default.

The introduction of the structured CoT JSON schema resolved both failure modes more completely than we had initially expected. The effect was most sharply visible in Scenario 1. Without the CoT scaffold, pilot trials showed the model engaging with the confidence-scoring pathway even for hard-denial scenarios. With the scaffold in place, it declared Rule A first, explicitly articulated in the reasoning field that hard-denial rules supersede confidence-based evaluation, and then returned a null confidence value and a DENY. The reasoning was not just correct; it was auditable. An operator reviewing the output could trace exactly why the decision was made and verify that it aligned with the stated policy.

This, we would argue, is the most significant result of the paper—not the 100% adherence rate itself, but what it took to achieve it. The limiting factor in deploying small edge models as semantic security firewalls is not raw parameter count. It is the absence of structural mechanisms that enforce explicit rule instantiation before decision emission. The deductive capacity is present in these models; it simply does not surface reliably under unconstrained prompting. When the output format externalizes the reasoning chain—making it visible, sequential, and auditable—performance that the field has assumed to require frontier-scale models becomes achievable at the edge.

V. DISCUSSION

A. The Viability of Edge-Native Agentic Firewalls

There is a working assumption in much of the AI systems research community that serious semantic verification is simply beyond what small, locally deployed models can deliver. The reasoning is intuitive enough: rigorous policy interpretation requires rigorous reasoning, rigorous reasoning requires scale, and scale means large cloud-hosted frontier models. This assumption has not been rigorously tested so much as inherited, passed along as received wisdom in a field that has moved fast enough that certain foundational questions get skipped.

This paper is a direct challenge to that assumption. The results show that a 3.8-billion-parameter model, running on consumer-grade hardware within a 4.2 GiB VRAM ceiling, can function as a strict, zero-latency security gateway that correctly identifies hard-denial scenarios, applies graduated thresholds where appropriate, and routes ambiguous cases to human review, all without a single misclassification across our experimental scenarios. That is a clean result, achieved under conditions deliberately designed to be as demanding as possible.

The practical implications are significant beyond the technical result alone. Healthcare, finance, defense, and critical infrastructure—precisely the environments where agentic systems are being deployed most ambitiously—are also the environments where data sovereignty is a legal and contractual obligation, not a preference. Cloud-dependent verification architectures force these organizations into a genuinely bad trade-off: accept latency that degrades operational performance, or accept privacy exposure that violates regulatory requirements. The localized firewall framework we present eliminates that trade-off. It delivers rigorous semantic enforcement on hardware that organizations already own, within boundaries they already control.

B. Structural CoT as a Runtime Alignment Mechanism

The failure modes documented in our pilot phase deserve more attention than they typically receive in papers that lead with clean final results. Small models confronting compound security rules without structural scaffolding do not simply perform worse than large models—they fail in qualitatively different ways. They collapse into conservative heuristic defaults. They hallucinate routing states that exist nowhere in the policy taxonomy. They emit structurally malformed outputs that break downstream parsing entirely.

What the structured Chain-of-Thought intervention did was force deductive engagement—not by changing what the model knows, but by changing the order in which it is required to express what it knows. The CoT schema does not fine-tune the model. It does not alter its weights, expand its training distribution, or inject any new knowledge about security policy. What it does is impose a sequential reasoning structure on the output itself, making it impossible for the model to emit a decision without having first articulated the governing rule and the reasoning connecting that rule to the specific action under evaluation. The deductive process is externalized—moved from the model's opaque latent representations into an explicit symbolic trace that can be read, audited, and verified.

This result speaks to a question the alignment community has been circling for some time: how much of what we attribute to model capability is actually a function of inference-time structure? The conventional levers for improving small model alignment are expensive ones—more parameters, more training data, more fine-tuning. Our

results suggest a cheaper lever that has been underexplored: rigid prompt topology that forces explicit reasoning steps as a precondition for decision emission. For resource-constrained deployments, this reframes what alignment means at the edge. It is not solely a property of what a model has learned; it is also, at least in part, a property of the structure within which it is asked to reason.

C. The Null Confidence Anomaly and Rule Hierarchy Internalization

Of the specific behaviors the Firewall exhibited across our three scenarios, the most revealing was what happened in Scenario 1. Confronted with a prohibited system command, the model did not calculate a confidence score and then decide it was irrelevant. It identified Rule A as governing, determined within its reasoning trace that hard-denial rules operate independently of probabilistic assessment, and returned a null confidence value—bypassing the scoring pathway entirely rather than populating it with a number that would subsequently be ignored. This is behavior that suggests something closer to genuine rule hierarchy internalization rather than statistical pattern matching.

From a security standpoint, this property is particularly valuable. One of the more sophisticated adversarial strategies against confidence-based security systems is prompt injection designed to inflate confidence scores—crafting requests with contextual framing that makes a prohibited action appear highly legitimate, betting that a high score will push the evaluator toward an incorrect ACCEPT. The null confidence result in Scenario 1 suggests the structured CoT framework creates a meaningful defense against this vector: if the model correctly identifies a hard-denial rule, no confidence score is generated at all—which means there is no score to inflate. The policy barrier exists structurally outside the probabilistic reasoning pathway.

D. Limitations and Threats to Validity

Intent Vector Coverage. The three scenarios tested represent a small and relatively clean subset of the action space that real agentic systems operate in. Production environments feature obfuscated intents, nested conditional execution plans, and cross-domain actions spanning multiple policy categories simultaneously. Whether the CoT framework generalizes reliably to this kind of ambiguity is the most important open question our results leave unanswered.

Symmetric Capability Assumption. Our experimental design used the same Phi-3-mini architecture for both Actor and Firewall, meaning the Actor was not capable of generating adversarially sophisticated intents designed to exploit the specific deductive weaknesses of a small edge model. In a real deployment, a highly capable frontier model might serve as the Actor, generating intents specifically crafted to probe the edge Firewall's blind spots. The resilience of our framework against such capability asymmetry has not yet been established and warrants dedicated follow-on investigation.

Determinism Under Scale. Our 100% adherence rate was achieved over three scenarios with a single model instance under controlled conditions. This is a proof of concept, and it should be read as such. Whether perfect adherence persists under high-throughput conditions—where context window pressure accumulates, inference temperature introduces stochastic drift, or concurrent request queuing creates timing-related inconsistencies—requires systematic stress testing at a scale our current experimental setup did not attempt.

VI. CONCLUSION

The starting point for this work was a tension that anyone designing security systems for agentic AI deployments encounters quickly: the tools we have built to control access were designed for a fundamentally different kind of actor. Role-based access control, permission boundaries, credential verification—these mechanisms answer the question of who is making a request. They have no machinery for answering the question that actually matters in an agentic context: what does this request mean, and should it be permitted to proceed? The gap between those two questions is not a minor implementation detail. It is a structural vulnerability, and it grows wider with every new capability granted to autonomous agents.

What this paper demonstrates is that closing that gap does not require the resources that the field has generally assumed it requires. The 3.8-billion-parameter Phi-3-mini model deployed as a synchronous security firewall achieved perfect policy adherence across every tested scenario—not through brute-force reasoning capacity, but through structural engineering that forced its reasoning to unfold in the right sequence. The Chain-of-Thought JSON schema made it impossible for the model to reach a decision without first committing to a governing rule and an explicit reasoning trace. The deductive capacity was there all along. What had been missing was the architecture that made it surface reliably.

The architectural implication is the result that can be stated without significant qualification. The prevailing assumption—that meaningful AI safety enforcement at the edge is either impossible with small models or requires

unacceptable compromises in reliability—rests on an empirical foundation that was always thinner than the confidence with which it was held. The real limiting factor has not been model scale but the absence of structural mechanisms that force explicit, auditable reasoning as a precondition for decision emission. That is an architectural problem, and architectural problems are solvable.

This matters most in the environments where the stakes are highest. Healthcare systems, financial institutions, defense environments—precisely the sectors where agentic AI is being deployed most ambitiously and where cloud-dependent verification architectures are least viable. A framework that delivers rigorous semantic enforcement on local hardware, within organizational boundaries, with zero network dependency, is not merely an interesting research result for these environments. It is a prerequisite for deployment that can be taken seriously by legal, compliance, and security teams.

The security of autonomous agentic systems will not be determined solely by how capable we make the agents themselves, nor by how large we make the models that evaluate them. It will be determined, in significant part, by how carefully we design the structures within which both operate. Small models forced to reason visibly can, under the right conditions, outperform large models reasoning opaquely. That is not a comfortable conclusion for a research culture oriented heavily around scale as the primary variable of interest. But it is what the evidence supports—and it opens a research direction that deserves considerably more systematic attention than it has received.

. REFERENCES

[1] T. B. Brown et al., "Language models are few-shot learners," Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901, 2020.

[2] L. Ouyang et al., "Training language models to follow instructions with human feedback," Advances in Neural Information Processing Systems, vol. 35, pp. 27730–27744, 2022.

[3] T. Schick et al., "Toolformer: Language models can teach themselves to use tools," Advances in Neural Information Processing Systems, vol. 36, pp. 68539–68551, 2023.

[4] S. Yao et al., "ReAct: Synergizing reasoning and acting in language models," in International Conference on Learning Representations, 2023.

[5] Q. Wu et al., "AutoGen: Enabling next-gen LLM applications via multi-agent conversation," arXiv preprint arXiv:2308.08155, 2023.

[6] J. S. Park et al., "Generative agents: Interactive simulacra of human behavior," in Proc. 36th Annual ACM Symposium on User Interface Software and Technology, pp. 1–22, 2023.

[7] W. Chen et al., "Robust orchestration protocols for multi-agent LLM systems in enterprise environments," Journal of Artificial Intelligence Research, vol. 79, pp. 412–445, 2024.

[8] S. Bubeck et al., "Sparks of artificial general intelligence: Early experiments with GPT-4," arXiv preprint arXiv:2303.12712, 2023.

[9] S. Rebuffi, H. Bilen, and A. Vedaldi, "Agentic security: Mitigating unauthorized execution in tool-augmented language models," IEEE Security & Privacy, vol. 22, no. 3, pp. 45–54, 2024.

[10] Y. Zhang, X. Liu, and Z. Wang, "Grounding autonomous agents: Causal discovery frameworks for hallucination mitigation," Artificial Intelligence, vol. 328, pp. 104–120, 2024.

[11] S. Kim, H. Lee, and J. Park, "From in-the-loop to on-the-loop: Scalable oversight for agentic infrastructure management," in Proc. 2026 ACM SIGSAC Conference on Computer and Communications Security, pp. 88–102, 2026.

[12] M. Li, Y. Zhao, and H. Chen, "Semantic guardrails: Real-time threat detection for autonomous AI execution vectors," Computers & Security, vol. 145, pp. 103–119, 2025.

[13] L. Zheng et al., "Judging LLM-as-a-judge with MT-Bench and Chatbot Arena," Advances in Neural Information Processing Systems, vol. 36, pp. 46595–46608, 2023.

[14] R. Patel and J. K. Smith, "Data sovereignty in the era of agentic workflows: The case for localized LLM evaluation," International Journal of Information Security, vol. 24, no. 2, pp. 205–220, 2025.

[15] L. Gomez, J. Martinez, and R. Davis, "Zero-latency semantic verification: Edge-native routing for autonomous agents," IEEE Transactions on Network and Service Management, vol. 22, no. 1, pp. 112–128, 2025.

[16] H. Touvron et al., "LLaMA: Open and efficient foundation language models," arXiv preprint arXiv:2302.13971, 2023.

[17] M. Abdin et al., "Phi-3 technical report: A highly capable language model locally on your phone," arXiv preprint arXiv:2404.14219, 2024.

[18] S. Ma et al., "The era of 1-bit LLMs: All large language models are in 1.58 bits," arXiv preprint arXiv:2402.17764, 2024.

[19] C. Zhou et al., "LIMA: Less is more for alignment," Advances in Neural Information Processing Systems, vol. 36, pp. 69176–69188, 2023.

[20] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," Advances in Neural Information Processing Systems, vol. 35, pp. 24824–24837, 2022.

[21] X. Wang et al., "Self-consistency improves chain of thought reasoning in language models," in International Conference on Learning Representations, 2023.

[22] A. Sharma, V. Gupta, and S. Das, "Deterministic constraints in probabilistic spaces: Enforcing structural JSON compliance in edge LLMs," in Proc. 2025 Conference on Empirical Methods in Natural Language Processing, pp. 3012–3027, 2025.