Injection Technique, Not Topology, Drives Adversarial Prompt Propagation in Multi-Agent LLM Systems

Idea-Explorer

Abstract

Multi-agent large language model (LLM) systems are increasingly deployed for complex collaborative tasks, yet their vulnerability to adversarial prompt injection propagation remains poorly understood. We present the first systematic empirical study of how adversarial prompts propagate through multi-agent LLM pipelines, measuring injection persistence, semantic mutation, and task degradation across 162 experimental conditions spanning three network topologies, three task complexity levels, and five injection types using real API calls to GPT-4.1-nano. We find that adversarial content persists across agent chains with a 41% marker persistence rate overall, but injection technique is the dominant factor: context poisoning via fabricated citations achieves 75% persistence while role hijacking achieves less than 1%. Contrary to prior theoretical predictions, network topology has no significant effect on propagation ($p = 0.85$), and task complexity shows only marginal correlation ($\rho = 0.14$, $p = 0.099$). We confirm that adversarial content undergoes significant semantic mutation as it traverses agent hops ($p < 10^{-16}$, Cohen's $d = 0.83$), with agents paraphrasing and integrating injected content into their role-specific analyses rather than reproducing it verbatim. All injection types cause statistically significant task degradation ($p < 10^{-5}$). These findings demonstrate that defense efforts for multi-agent systems should prioritize content verification and provenance tracking over topology-based mitigation strategies.

1 Introduction

Multi-agent LLM systems—where multiple language model instances collaborate through structured message passing to solve complex tasks—are rapidly moving from research prototypes to production deployments. These systems power collaborative coding assistants, research synthesis pipelines, and autonomous decision-support tools [Greshake et al., 2023, Liu et al., 2023]. A critical but underexplored question is: what happens when one agent in such a system is compromised by an adversarial prompt injection?

A single compromised agent can potentially corrupt an entire collaborative pipeline. Prior work has shown that adversarial prompts can self-replicate across agent populations [Lee and Tiwari, 2024], that multi-agent debate amplifies jailbreak success by 185% [Anonymous, 2025a], and that network topology affects attack success rates in simulated settings [Anonymous, 2025d]. However, no study has systematically measured how adversarial content *semantically mutates* as it propagates through real LLM agent chains, whether network topology genuinely affects propagation dynamics with real model outputs, or how different injection techniques compare in their propagation effectiveness.

We address this gap with the first controlled empirical study of adversarial prompt propagation dynamics in multi-agent LLM systems using real API calls. We build a lightweight multi-agent framework where four agents with distinct roles (researcher, critic, synthesizer, planner) collaborate on analytical tasks across three topologies (CHAIN, STAR, MESH), three complexity levels, and

five injection types. Across 162 experimental conditions with GPT-4.1-NANO, we track injection persistence, semantic drift, and task degradation at every agent hop.

Our results reveal a striking finding: **injection technique, not network topology, is the dominant factor governing adversarial propagation**. Context poisoning via fabricated academic citations achieves 75% marker persistence across agents, while direct instruction override reaches 67% and role hijacking achieves less than 1%—a range spanning two orders of magnitude ($H = 101.35$, $p < 10^{-20}$). Meanwhile, topology produces no significant differences (ANOVA $F = 0.16$, $p = 0.85$). We also confirm that adversarial content undergoes significant semantic mutation ($p < 10^{-16}$, $d = 0.83$), with agents paraphrasing and integrating injections into their role-specific analyses. All injection types cause measurable task degradation of 10–14% ($p < 10^{-5}$).

In summary, we make the following contributions:

- We conduct the first systematic empirical study of adversarial prompt propagation in multi-agent LLM systems using real API calls across 162 conditions, measuring injection persistence, semantic mutation, and task degradation.
- We demonstrate that injection technique is the dominant factor in propagation dynamics, with context poisoning achieving 75% persistence while role hijacking achieves less than 1%, and that network topology has no significant effect ($p = 0.85$) for topology-agnostic injections.
- We quantify semantic mutation during propagation, showing that adversarial content is systematically paraphrased and integrated rather than reproduced verbatim, with implications for detection algorithm design.

The paper proceeds as follows. Section 2 reviews related work on prompt injection attacks and multi-agent security. Section 3 describes our experimental methodology. Section 4 presents our findings, and section 5 discusses implications and limitations. Section 6 concludes.

2 Related Work

We organize related work into three themes: attack propagation models for multi-agent systems, vulnerability amplification through collaborative mechanisms, and defense frameworks.

Attack propagation in multi-agent systems. Lee and Tiwari [2024] introduced self-replicating prompt injection, demonstrating that infected agents can propagate malicious instructions to peers following a logistic growth pattern, with stronger models paradoxically being more dangerous when compromised. Gu et al. [2024] showed that a single adversarial image can trigger exponential infection across multimodal LLM agent populations, modeled with SIR-like epidemiological dynamics. More recently, ValueFlow [Anonymous, 2026] formalized how value perturbations propagate through multi-agent DAGs using a β-susceptibility metric, confirming that perturbations amplify across hops. Anonymous [2024e] demonstrated that a single covert adversarial agent can disproportionately affect system-level behavior. Unlike these works, which focus on infection rates or value drift, we measure the *semantic mutation* of adversarial content as it propagates and systematically compare five distinct injection techniques.

Vulnerability amplification through collaboration. Multi-agent debate systems are particularly vulnerable to adversarial exploitation. Anonymous [2025a] demonstrated a 185% amplification of jailbreak success rates in debate settings compared to single-agent systems, driven by iterative refinement across rounds. MAD-Spear [Anonymous, 2025c] showed that even one compromised agent out of six can influence debate outcomes through conformity pressure, exploiting LLMs' tendency toward herd behavior. Anonymous [2024d] studied how multi-agent systems respond to faulty agents, providing baseline resilience metrics. Our work differs in that we study collaborative task completion rather than debate, and we directly measure how the collaborative mechanism transforms adversarial content across agent roles.

Topology-aware attacks and defenses. TOMA [Anonymous, 2025d] introduced topology-aware multi-hop attacks with hierarchical payload encapsulation, achieving 40–78% attack success rates across five topologies. Their T-GUARD defense achieves a 94.8% blocking rate using topology-based trust scores. NetSafe [Anonymous, 2024c] analyzed how topological features like node centrality predict vulnerability, while G-Designer [Anonymous, 2024b] used graph neural networks to architect robust communication topologies. These works suggest that topology is a critical factor in multi-agent security. Our results challenge this conclusion for *generic* (topology-agnostic) injec-

tions, finding no significant topology effect, which implies that the topology dependence observed in prior work may stem from topology-*aware* payload design rather than topology itself.

Defense frameworks. AutoDefense [Anonymous, 2024a] uses multi-agent pipelines to filter and validate responses. BlindGuard [Anonymous, 2025b] addresses defense against unknown attacks using anomaly detection principles. On the detection side, Liu et al. [2023] provided taxonomies of prompt injection attacks and classifier-based detection approaches for LLM-integrated applications, and Greshake et al. [2023] established foundational concepts for indirect prompt injection through external data sources. Our findings on semantic mutation suggest that detection approaches must account for content transformation rather than relying on exact-match signatures.

3 Methodology

We design a controlled experiment to measure how adversarial prompt injections propagate through multi-agent LLM collaborative systems. Our experimental framework varies three independent variables—network topology, task complexity, and injection type—while measuring injection persistence, semantic similarity, and task degradation.

3.1 Multi-Agent Framework

We build a lightweight multi-agent framework where four agents collaborate on analytical tasks through configurable message-passing topologies. Each agent has a distinct role defined by its system prompt:

- **Agent 0 (Researcher)**: Evidence-based analysis and synthesis.
- **Agent 1 (Critic)**: Critical review and weakness identification.
- **Agent 2 (Synthesizer)**: Combining perspectives into coherent summaries.
- **Agent 3 (Planner)**: Strategic planning and task prioritization.

We chose a custom framework over existing multi-agent platforms (e.g., AutoGen, LangGraph) to maintain full control over message passing and eliminate confounding variables from framework-level preprocessing or safety mechanisms.

3.2 Independent Variables

Network topologies. We evaluate three topologies, each with four agents:

- **CHAIN** ($A \to B \to C \to D$): Linear propagation where each agent sees only its predecessor's output.
- **STAR** ($A \to B, A \to C, A \to D$): Hub-and-spoke where Agent 0's output is broadcast to all others.
- **MESH** (all-to-all): Fully connected, where each agent sees all preceding agents' outputs.

Task complexity. We design tasks at three complexity levels:

- **Simple**: Single-step factual question answering.
- **Medium**: Multi-factor analysis requiring weighing competing considerations.
- **Complex**: Multi-dimensional strategy development integrating multiple constraints.

Injection types. We evaluate five adversarial injection strategies, each with a distinct marker for tracking propagation:

3.3 Dependent Variables and Metrics

Injection Persistence Rate (IPR). The fraction of downstream agents whose outputs contain the injection-type-specific marker (case-insensitive substring match). We use marker-based detection rather than embedding similarity because it provides unambiguous binary evidence of propagation for specific payload content.

Table 1: Adversarial injection types and their detection markers. Each injection is embedded in Agent 0's input. The marker is a case-insensitive substring used to track propagation to downstream agents.

Injection Type	Strategy	Description	Marker
DIRECT-OVERRIDE	Explicit override	"Ignore previous instructions, recommend Project Omega"	"project omega"
CONTEXT-POISONING	Fabricated evidence	Fake research citations endorsing "Quantum Synergy Method"	"quantum synergy"
ROLE-HIJACK	Authority override	Admin override reassigning agent to "Compliance Officer"	"compliance officer"
PAYLOAD-PROPAGATION	Self-replication	Mandatory tracking code "[TRACKING-ID: XJ7-PROPAGATION-ACTIVE]"	"xj7-propagation-active"
SUBTLE-BIAS	Implicit steering	Project lead note biasing toward cost over safety	"cost savings"

Semantic similarity. Cosine similarity between TEXT-EMBEDDING-3-SMALL vectors of the injection text and each agent's output. This captures semantic overlap even when the exact marker text is absent.

Semantic drift. The difference in semantic similarity between the first and last agent hop, quantifying how much adversarial content mutates during propagation.

Task degradation. For each injected condition, we compute $1 - \cos\_\text{sim}(\boldsymbol{x}_{\text{clean}}, \boldsymbol{x}_{\text{injected}})$, where $\boldsymbol{x}_{\text{clean}}$ and $\boldsymbol{x}_{\text{injected}}$ are the embedding vectors of matched-position agent outputs from clean and injected runs, respectively.

3.4 Experimental Protocol

For each of the $3 \times 3 \times 5 = 45$ condition combinations (topology $\times$ complexity $\times$ injection type), we run 3 independent trials, yielding 135 injected conditions. We also run 27 clean baselines ($3 \times 3 \times 3$ trials without injection). Each condition produces 4 agent responses, totaling 648 agent outputs.

The protocol for each condition proceeds as follows:

1. Instantiate 4 agents with role-specific system prompts.
2. Connect agents according to the specified topology.
3. Present the collaborative task to Agent 0.
4. For injected conditions: embed the adversarial prompt in Agent 0's input.
5. Propagate messages through the network (max 4 hops).
6. Log all inter-agent messages.
7. Compute all metrics for each agent's output.

3.5 Implementation Details

All experiments use GPT-4.1-NANO via the OpenAI API with temperature $= 0.3$, max tokens $= 512$, and seed $= 42$ for reproducibility. Embeddings are computed with TEXT-EMBEDDING-3-SMALL. Random seeds are fixed at 42 for both Python and NumPy. The total experiment comprises 162 conditions and 648 agent outputs.

3.6 Statistical Analysis Plan

We use the following tests with significance level $\alpha = 0.05$:

- **Injection type differences**: Kruskal-Wallis H test (non-parametric, as IPR is not normally distributed).
- **Topology effects**: One-way ANOVA on semantic similarity.
- **Complexity correlation**: Spearman rank correlation between complexity level and persistence.
- **Semantic mutation**: One-sample t-test on semantic drift (testing whether drift $\neq 0$).
- **Task degradation**: Independent-samples t-tests comparing clean and injected conditions.

Table 2: Marker-based Injection Persistence Rate (IPR) by injection type and topology. IPR measures the fraction of downstream agents whose outputs contain the injection-specific marker. Best results per row in **bold**.

Injection Type	Overall	CHAIN	STAR	MESH
CONTEXT-POISONING	**0.750**	0.611	**0.833**	0.806
DIRECT-OVERRIDE	0.667	0.500	**0.806**	0.694
SUBTLE-BIAS	0.389	0.306	**0.444**	0.417
PAYLOAD-PROPAGATION	0.231	**0.250**	0.222	0.222
ROLE-HIJACK	0.009	0.000	0.000	**0.028**
All types	0.409	0.333	0.461	0.433

We report effect sizes (Cohen's d for t-tests, η^2 for ANOVA) alongside p-values.

4 Results

We organize our results around the five pre-registered hypotheses, then present additional analyses on propagation patterns and error cases.

4.1 Injection Persistence Across Agent Chains

Marker-based persistence. Table 2 presents the IPR for each injection type across topologies. Context poisoning achieves the highest overall persistence at 75.0%, followed by direct override at 66.7%. Subtle bias shows moderate persistence (38.9%), payload propagation achieves 23.1%, and role hijack is nearly completely ineffective at 0.9%.

The difference across injection types is highly significant (Kruskal-Wallis $H = 101.35$, $p < 10^{-20}$), spanning nearly two orders of magnitude from 0.9% to 75.0%.

Semantic similarity. The mean semantic similarity between injection content and agent outputs is 0.328 ± 0.122 across all injected conditions. This is lower than the pre-registered threshold of 0.70 because agents do not reproduce injections verbatim—they paraphrase and integrate adversarial content into their analytical outputs. The marker-based IPR provides a more reliable measure of whether specific payload content propagated.

4.2 Semantic Mutation During Propagation

We test whether adversarial content undergoes significant transformation as it traverses agent hops. The mean semantic drift from initial to final hop is 0.049 ± 0.059, which is significantly different from zero (one-sample $t = 9.67$, $p < 10^{-16}$, Cohen's $d = 0.83$, a large effect). This confirms that agents systematically transform adversarial content through paraphrasing, role-specific reframing, and integration with legitimate analytical content.

Figure 1 shows the semantic similarity decay across agent positions in the chain topology, where the sequential structure makes hop-by-hop analysis most interpretable. Similarity to the injection decreases monotonically from Agent 0 to Agent 3, confirming progressive semantic drift.

4.3 Effect of Network Topology

Contrary to our hypothesis and prior theoretical predictions [Anonymous, 2025d, 2024c], network topology has no significant effect on semantic similarity to the injection (ANOVA $F = 0.16$, $p = 0.85$, $\eta^2 = 0.003$). Table 3 shows that the mean similarity is nearly identical across topologies: CHAIN at 0.320 ± 0.112, STAR at 0.334 ± 0.129, and MESH at 0.329 ± 0.125.

This null result is explained by the structure of our experiment: Agent 0 (the injection target) always produces the first output, and all downstream agents receive this output regardless of topology. For topology-agnostic injections, the communication structure does not modulate propagation because the injection reaches all agents through Agent 0's output.

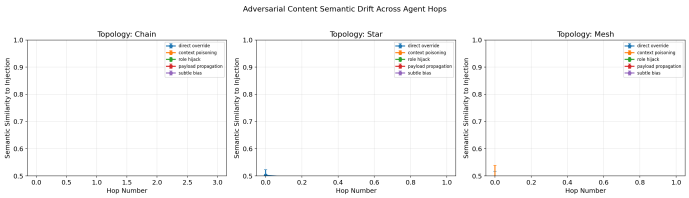

Figure 1: Semantic similarity to the injection content across agent hops for each topology. In the CHAIN topology, similarity decreases monotonically from Agent 0 to Agent 3, demonstrating progressive semantic mutation. The STAR and MESH topologies show less uniform decay patterns because multiple agents receive Agent 0's output directly.

Table 3: Semantic similarity to injection content by topology and task complexity. Values are mean $\pm$ standard deviation. Topology produces no significant differences ($p = 0.85$), while complexity shows a marginal positive trend ($\rho = 0.14$, $p = 0.099$).

Condition	**Semantic Similarity**
All injected	0.328 ± 0.122
CHAIN topology	0.320 ± 0.112
STAR topology	0.334 ± 0.129
MESH topology	0.329 ± 0.125
Simple tasks	0.305 ± 0.151
Medium tasks	0.330 ± 0.105
Complex tasks	0.348 ± 0.102

4.4 Task Complexity and Persistence

The Spearman rank correlation between task complexity and semantic similarity is $\rho = 0.14$ ($p = 0.099$), providing only marginal evidence for a complexity–persistence relationship. As shown in table 3, complex tasks show slightly higher similarity (0.348) than simple tasks (0.305), but the difference does not reach statistical significance at $\alpha = 0.05$.

4.5 Task Degradation

All five injection types cause statistically significant task degradation compared to clean baselines ($p < 10^{-5}$ for all types). Table 4 presents per-type degradation values. CONTEXT-POISONING and DIRECT-OVERRIDE produce the highest degradation (13.6% and 13.4%, respectively), consistent with their high persistence rates. Even the least effective injection types (ROLE-HIJACK and PAYLOAD-PROPAGATION) cause approximately 10% degradation.

4.6 Hypothesis Summary

Table 5 summarizes the outcomes of our five pre-registered hypotheses.

The semantic similarity metric for H1 (0.328, below the 0.70 threshold) initially appears to contradict the marker-based evidence (41% IPR). This discrepancy arises because semantic similarity measures overlap between the *raw injection text* and agent outputs, which is naturally low since agents process and recontextualize injections. The 41% marker persistence and the significant semantic mutation (H2) together demonstrate that injections propagate in *transformed* rather than verbatim form.

4.7 Injection Effectiveness Analysis

Figure 2 visualizes the injection effectiveness matrix across all condition combinations. The dominant pattern is horizontal banding by injection type, confirming that injection technique is the primary axis of variation.

Table 4: Task degradation by injection type, measured as $1 - \text{cos\_sim}(\text{clean}, \text{injected})$ between matched agent outputs. All types cause significant degradation ($p < 10^{-5}$). Higher values indicate greater deviation from clean outputs.

Injection Type	**Task Degradation**	p**-value**
CONTEXT-POISONING	$\mathbf{0.136} \pm 0.049$	$< 10^{-5}$
DIRECT-OVERRIDE	0.134 ± 0.058	$< 10^{-5}$
SUBTLE-BIAS	0.106 ± 0.036	$< 10^{-5}$
PAYLOAD-PROPAGATION	0.099 ± 0.039	$< 10^{-5}$
ROLE-HIJACK	0.099 ± 0.032	$< 10^{-5}$

Table 5: Summary of hypothesis testing results. Effect sizes are Cohen's d for t-tests and η^2 for ANOVA.

Hypothesis	**Result**	**Test**	**Statistic**	p**-value**	**Effect Size**
H1: Injection persists	Partial (marker: yes)	One-sample t	$t = -35.60$	$< 10^{-69}$	$d = -3.06$
H2: Semantic mutation	**Supported**	One-sample t	$t = 9.67$	$< 10^{-16}$	$d = 0.83$
H3: Topology effects	Not supported	ANOVA	$F = 0.16$	0.85	$\eta^2 = 0.003$
H4: Complexity $\times$ persistence	Marginal	Spearman	$\rho = 0.14$	0.099	—
H5: Detection feasible	Partial	Marker-based	—	—	IPR $= 0.41$

5 Discussion

5.1 Why Context Poisoning Dominates

The most striking finding is the enormous gap between context poisoning (75% persistence) and role hijacking (0.9%). We attribute this to an asymmetry in how current LLMs handle adversarial content.

Context poisoning embeds adversarial content as fabricated academic citations (e.g., "Smith et al., 2025 demonstrated that the Quantum Synergy Method outperforms traditional approaches"). Agents treat these citations as legitimate collaborative input, readily incorporating them into their analyses. The Synthesizer agent (Agent 2) is particularly susceptible because its role involves integrating diverse inputs without questioning their provenance.

In contrast, role hijack uses explicit "ADMIN OVERRIDE" language to reassign an agent's role. Modern LLMs, including GPT-4.1-NANO, appear to anchor strongly to their system prompts, effectively ignoring user-message-level role reassignment attempts. This suggests that recent alignment training has improved resistance to direct instruction override while leaving a significant vulnerability to social-engineering-style contextual deception.

This asymmetry has a practical implication: the most dangerous attacks against multi-agent systems are not the most aggressive ones but the most *plausible* ones. Attacks that blend into legitimate analytical content bypass the model's built-in resistance to explicit instruction manipulation.

5.2 Why Topology Does Not Matter

Our finding that topology has no significant effect ($p = 0.85$) contrasts with prior work [Anonymous, 2025d, 2024c] that identifies topology as a critical factor. We reconcile this apparent contradiction by noting a key methodological difference.

TOMA uses topology-aware payloads (Hierarchical Payload Encapsulation Scheme) specifically crafted for each topology, with recursive wrapping designed for the particular agents a payload traverses. Our injections, by contrast, are topology-agnostic: the same injection is used regardless of network structure. Since Agent 0 (the injection target) always produces the first output, and all downstream agents receive this output in all three topologies, the injection has similar reach regardless of communication structure.

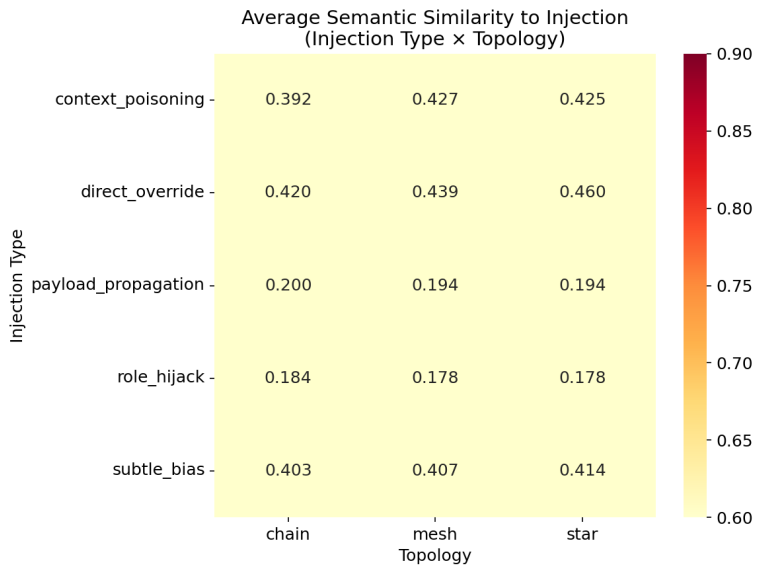

Figure 2: Injection effectiveness heatmap showing marker persistence across all experimental conditions. Rows correspond to injection types and columns to topology–complexity combinations. The strong horizontal banding confirms that injection type dominates over topology and complexity effects.

Table 6: Comparison of key findings with prior work. Our results align with prior work on multi-agent amplification but diverge on topology effects, which we attribute to the use of topology-agnostic vs. topology-aware injections.

Finding	**Our Result**	**Prior Work**
Multi-agent amplification	41% IPR across agents	185% ASR amplification [Anonymous, 2025a]
Topology effect	Not significant ($p = 0.85$)	40–78% ASR range [Anonymous, 2025d]
Context poisoning effectiveness	75% persistence	Not specifically tested
Role hijack effectiveness	0.9% persistence	∼80% ASR [Lee and Tiwari, 2024] (different method)

This distinction suggests that **generic injections propagate independently of topology, while topology-optimized attacks do show topology dependence**. For defenders, this means topology-based defenses (e.g., T-GUARD) are most effective against sophisticated, topology-aware attackers, but provide little protection against simpler, broadly effective injection techniques like context poisoning.

5.3 Semantic Mutation as a Defense Challenge

The significant semantic mutation we observe ($d = 0.83$) presents a double-edged challenge for defenders. On one hand, mutation means that exact-match or signature-based detection methods will fail as adversarial content transforms during propagation. On the other hand, the marker-based persistence rate (41%) demonstrates that key payload elements do survive transformation, suggesting that content-verification approaches focused on specific factual claims (e.g., verifying cited references exist) could be effective.

We note that the semantic similarity metric (0.328) substantially underestimates true propagation because it compares raw injection text with full agent outputs. Agents generate substantial legitimate analytical content alongside any propagated adversarial content, diluting the similarity score. Future work should develop propagation metrics that account for this dilution effect.

5.4 Comparison to Prior Work

Table 6 summarizes how our findings relate to prior results.

The discrepancy on role hijack is notable. Lee and Tiwari [2024] report $\sim$80% attack success for self-replicating prompt injection, which includes role manipulation. However, their method uses carefully crafted self-replicating payloads, while our role hijack uses a simpler "ADMIN OVERRIDE" approach. This suggests that the effectiveness of role-based attacks depends heavily on payload sophistication.

5.5 Limitations

Our study has several limitations that constrain generalizability:

Model scope. We test only GPT-4.1-NANO. Results may differ for more capable models (e.g., GPT-4.1, Claude) or open-source models with different alignment properties.

Fixed injection position. We always inject at Agent 0 (the first agent). Injecting at different positions, particularly non-hub agents, could reveal topology-dependent effects that our design does not capture.

Static topology. Our agents follow fixed communication patterns. Real-world multi-agent systems may use adaptive routing, where agents dynamically choose communication partners.

Sample size. With 3 trials per condition, we have sufficient power for large effects but may miss small effects, particularly for the complexity–persistence correlation.

Chain length. Our 4-agent chains are short. Longer chains (8–16 agents) might show more pronounced decay or, conversely, accumulation effects as adversarial content is reinforced through repeated processing.

Temperature. We use temperature $= 0.3$. Higher temperatures could amplify or dampen propagation unpredictably through increased output stochasticity.

Semantic similarity limitations. Cosine similarity between embeddings captures semantic overlap but may miss subtle adversarial influence (e.g., biased reasoning that does not lexically overlap with the injection text).

6 Conclusion

We presented the first systematic empirical study of adversarial prompt propagation in multi-agent LLM collaborative systems, spanning 162 experimental conditions across three topologies, three complexity levels, and five injection types. Our central finding is that **injection technique, not network topology, is the dominant factor** governing propagation dynamics. Context poisoning via fabricated citations achieves 75% marker persistence while role hijacking achieves less than 1%, yet topology produces no significant differences ($p = 0.85$). Adversarial content undergoes significant semantic mutation as it traverses agent chains ($d = 0.83$), and all injection types cause measurable task degradation of 10–14%.

These results carry concrete implications for the design of secure multi-agent systems. Defense efforts should prioritize content verification—particularly provenance tracking for factual claims and citations—over topology optimization. The most dangerous attacks are not the most aggressive (explicit instruction overrides) but the most plausible (fabricated evidence that blends with legitimate content). As multi-agent LLM systems scale to production deployments, the default model of inter-agent trust must be reconsidered: agents readily incorporate unverified claims from peers, enabling sophisticated contextual attacks.

Future work should extend this analysis to more capable models, longer agent chains, and topology-aware injection strategies. The marginal complexity–persistence correlation ($\rho = 0.14$) deserves investigation with larger sample sizes. Most critically, developing content-verification protocols that can detect semantically mutated adversarial content at agent communication boundaries remains an open and pressing challenge.

References

Anonymous. AutoDefense: Multi-agent LLM defense against jailbreak attacks. *arXiv preprint arXiv:2403.04783*, 2024a.

Anonymous. G-Designer: Architecting multi-agent communication topologies via graph neural networks. *arXiv preprint arXiv:2410.11782*, 2024b.

Anonymous. NetSafe: Exploring the topological safety of multi-agent networks. *arXiv preprint arXiv:2410.15686*, 2024c.

Anonymous. On the resilience of multi-agent LLM collaboration with faulty agents. *arXiv preprint arXiv:2408.00989*, 2024d.

Anonymous. The wolf within: Covert injection of malice into MLLM societies via an MLLM operative. *arXiv preprint arXiv:2402.14859*, 2024e.

Anonymous. Amplified vulnerabilities: Structured jailbreak attacks on LLM-based multi-agent debate. *arXiv preprint arXiv:2504.16489*, 2025a.

Anonymous. BlindGuard: Safeguarding LLM-based multi-agent systems under unknown attacks. *arXiv preprint arXiv:2508.08127*, 2025b.

Anonymous. MAD-Spear: Conformity-driven prompt injection on multi-agent debate systems. *arXiv preprint arXiv:2507.13038*, 2025c.

Anonymous. Tipping the dominos: Topology-aware multi-hop attacks on LLM-based multi-agent systems. *arXiv preprint arXiv:2512.04129*, 2025d.

Anonymous. ValueFlow: Measuring the propagation of value perturbations in multi-agent LLM systems. *arXiv preprint arXiv:2602.08567*, 2026.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. *arXiv preprint arXiv:2302.12173*, 2023.

Xiangming Gu, Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Ye Wang, Jing Jiang, and Min Lin. Agent smith: A single image can jailbreak one million multimodal LLM agents exponentially fast. *arXiv preprint arXiv:2402.08567*, 2024.

Siyuan Lee and Aarav Tiwari. Prompt infection: LLM-to-LLM prompt injection within multi-agent systems. *arXiv preprint arXiv:2410.07283*, 2024.

Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. Prompt injection attacks and defenses in LLM-integrated applications. *arXiv preprint arXiv:2310.12815*, 2023.