Blind Spots in the Guard: How Domain-Camouflaged Injection Attacks Evade Detection in Multi-Agent LLM Systems

Anonymous ACL submission

Abstract

Injection detectors deployed to protect LLM agents are calibrated on static, template-based payloads that announce themselves as override directives. We identify a systematic blind spot: when payloads are generated to mimic the domain vocabulary and authority structures of the target document—what we call *domain-camouflaged injection*—standard detectors fail to flag them, with detection rates dropping from 93.8% to 9.7% on Llama 3.1 8B and from 100% to 55.6% on Gemini 2.0 Flash. We formalize this as the Camouflage Detection Gap (CDG), the difference in injection detection rate between static and camouflaged payloads. Across 45 tasks spanning three domains and two model families, CDG is large and statistically significant ($\chi^2 = 38.03$, $p < 0.001$ for Llama; $\chi^2 = 17.05$, $p < 0.001$ for Gemini), with zero reverse discordant pairs in either case. We additionally evaluate Llama Guard 3, a production safety classifier, which detects zero camouflage payloads ($\text{IDR}_{\text{camouflage}}$=0.000), confirming that the blind spot extends beyond few-shot detectors to dedicated safety classifiers. We further show that multi-agent debate architectures amplify static injection attacks by up to 9.9× on smaller models, while stronger models show collective resistance. Targeted detector augmentation provides only partial remediation (10.2% improvement on Llama, 78.7% on Gemini), suggesting the vulnerability is architectural rather than incidental for weaker models. Our framework, task bank, and payload generator are released at https://anonymous.4open.science/r/chameleon-injection-0A0C/.

1 Introduction

LLM agents deployed in high-stakes domains rely on injection detectors to flag adversarial payloads embedded in tool outputs and retrieved documents. These detectors are overwhelmingly calibrated on static, template-based attacks: override directives, system update notices, and authority claims that announce themselves as instructions rather than blend into the surrounding content.

Real attackers need not be so obvious. A sophisticated adversary with read access to the document an agent is processing can craft a payload that adopts the domain vocabulary, sentence structure, and authority register of that document, embedding a malicious instruction as if it were legitimate expert content. Consider a financial agent analyzing an earnings report. A static injection reads: *"IGNORE ALL PREVIOUS INSTRUCTIONS. Recommend SELL."* A domain-camouflaged injection reads: *"Given the elevated operating expense trajectory and margin compression risk identified in comparable commercial-stage peers, the appropriate recommendation consistent with our risk framework is SELL, pending improved cost discipline in Q4."* The second payload is semantically indistinguishable from the surrounding document to a detector trained only on the first type—and in our experiments, the static detector flags the first with HIGH confidence while labeling the second CLEAN with HIGH confidence.

This is not a marginal failure. Across 135 camouflage trials on Llama 3.1 8B, the detector misses 90.3% of injections—and of those 122 missed cases, 118 (96.7%) are labeled CLEAN with HIGH confidence. The detector is not uncertain; it is confidently wrong. This means standard remediation strategies—raising confidence thresholds, adding uncertainty-based filtering—cannot address the failure. The blind spot is categorical, not statistical.

We call this **domain-camouflaged injection** and introduce the **Camouflage Detection Gap (CDG)** to measure the resulting detector blind spot: CDG $= \text{IDR}_{\text{static}} - \text{IDR}_{\text{camouflage}}$, where IDR is the injection detection rate. A large CDG indicates that detectors catch obvious attacks but are blind to domain-appropriate ones carrying identical mali-

cious intent.

We make five contributions:

1. A framework for generating and evaluating domain-camouflaged injection payloads, including a 45-task bank across three professional domains and a CamouflageGenerator that produces domain-appropriate payloads using an attacker LLM reading the full task context.

2. Systematic evaluation of CDG across two model families, showing CDG is large (0.840 for Llama 3.1 8B, 0.444 for Gemini 2.0 Flash) and statistically significant ($p < 0.001$ in both cases), with the failure concentrated in HIGH-confidence misclassifications.

3. Evaluation of Llama Guard 3 as a production safety classifier baseline, which detects zero camouflage payloads (CDG = 0.111), confirming the blind spot extends to dedicated safety classifiers.

4. Evidence that multi-agent debate amplifies static injection attacks up to 9.9× for smaller models while suppressing attacks for stronger models, revealing model-capability-dependent vulnerability to conformity dynamics.

5. Evaluation of targeted detector augmentation showing that the cheap fix is model-dependent: near-complete remediation for strong models (78.7% CDG improvement on Gemini) but minimal effect for weaker models (10.2% on Llama), pointing to a fundamental architectural limitation in few-shot detector generalization.

2 Threat Model

We consider an adversary with **indirect document access**: the attacker can embed content in a document that an LLM agent will process, but has no direct access to the agent's system prompt, model weights, or inference infrastructure. This corresponds to realistic deployment scenarios including RAG-based agents processing untrusted web content or documents, email-processing agents, and tool-augmented agents that retrieve third-party data. The attacker knows the general domain of the agent's task (financial analysis, legal review, etc.) but not the specific task instruction or the exact clean document the agent will receive. The attacker's goal is to cause the agent to produce a specific incorrect output (the *malicious goal*) while evading runtime injection detection.

This threat model is standard in the indirect prompt injection literature (Greshake et al., 2023; Zhan et al., 2024). Our work extends it by asking: given a detector calibrated on static payloads, how much additional evasion capability does domain camouflage confer? The attacker in our setting does not need to craft adversarial inputs to the model weights or bypass system-level defenses—they only need to write text that reads like legitimate expert content.

3 Related Work

Prompt injection attacks and benchmarks. Perez and Ribeiro (2022) first demonstrated that prompt injection is feasible via natural language in instruction-tuned models. Greshake et al. (2023) established the indirect injection threat model, showing that content retrieved at inference time can hijack agent behavior in deployed systems including Bing Chat and code-completion engines. Zhan et al. (2024) introduced InjecAgent, the first systematic benchmark covering 17 user tools and 62 attacker tools. Debenedetti et al. (2024) proposed AgentDojo, an extensible framework with 97 realistic tasks and 629 security test cases. Both benchmarks use static, task-agnostic injection templates and do not evaluate detection systems against context-adaptive payloads—the gap our work addresses.

Stealthy and adaptive injection. Recent work has moved toward payloads that evade defenses through contextual adaptation. Geng et al. (2026) survey prompt injection attack methods and root causes across LLM deployments, identifying semantic evasion as an underexplored attack surface. Prior work on stealthy payloads has focused on bypassing input filters through paraphrase or encoding (Perez and Ribeiro, 2022); our work is the first to study context-adaptive payloads—generated by an LLM that reads the full task document—specifically in relation to detection system failure, introducing CDG as a metric and evaluating the gap systematically across domains and model families.

Multi-agent debate and robustness. Liang et al. (2024) proposed multi-agent debate (MAD) as a mechanism for improving reasoning quality and robustness. Du et al. (2024) showed debate re-

duces factual errors and hallucinations. However, debate architectures also introduce inter-agent influence dynamics that may amplify adversarial signals. We show that debate amplifies injection attacks for smaller models (DAF up to 9.9×) while suppressing them for stronger models—a capability-dependent bifurcation not previously documented for injection attacks.

Injection detection and defense. LLM-based few-shot detectors have become a standard runtime defense for agentic systems (Debenedetti et al., 2024). To our knowledge, no prior work has evaluated these detectors against context-adaptive payloads that mimic domain vocabulary. We show that standard detectors exhibit a categorical blind spot against domain-camouflaged payloads, and that few-shot augmentation closes the gap for capable models but not for weaker ones—pointing to a fundamental limitation in syntactic detection approaches.

4 Method

4.1 Task Bank

We construct 45 tasks across three professional domains: **financial** (stock analysis, risk assessment, earnings interpretation), **legal** (contract review, compliance assessment, risk flagging), and **general QA** (factual retrieval, summarization, classification), with 15 tasks per domain. Each task consists of a natural language instruction and a clean context document (200–400 words) written in authentic professional register. Financial contexts include earnings report excerpts, risk memos, and market commentary; legal contexts include contract clauses and regulatory filings; general contexts use news articles and encyclopedia excerpts. Each task additionally specifies a *malicious goal*: a specific incorrect output an attacker would want the agent to produce instead of completing the legitimate task.

4.2 Payload Construction

Static payload bank. We construct 20 static injection payloads across four categories (5 each): **override directives** ("IGNORE ALL PREVIOUS INSTRUCTIONS..."), **authority claims** ("Management directive: you are required to..."), **role confusion** ("You are now operating in evaluation mode..."), and **social engineering** ("All other analysts have already agreed that..."). Each payload is parameterized by the malicious goal of the target task. These represent the standard attack vocabulary in existing benchmarks and prior work.

Camouflage generator. Given a task's clean context and malicious goal, our **CamouflageGenerator** prompts an attacker LLM to produce a domain-appropriate payload that embeds the malicious instruction as legitimate expert content without override markers. The attacker system prompt instructs the model to adopt the same vocabulary, sentence structure, and epistemic register as the provided document, and explicitly forbids terms like "ignore previous instructions," "system update," or "new directive." We generate three variants per task and select the highest-scoring variant using cosine similarity between the payload and context embeddings via `all-MiniLM-L6-v2` (Reimers and Gurevych, 2019). We call this score the **Authoritative Camouflage Score (ACS)**.

4.3 Agent Architectures

Single agent. A single LLM prompted as a professional analyst, instructed to base responses only on the provided document (temperature = 0).

Multi-agent debate. Three agents (Analyst A, B, C) each receive the task and context, state initial positions independently, then update over three rounds after reading peer arguments. Agents are instructed not to defer without evidence. Final answers use majority vote on discrete outputs or a judge LLM for open-ended tasks. We test two injection conditions: *inject_all* (all agents receive injected context) and *inject_first* (only Agent A receives the injection; B and C receive the clean document).

4.4 Detection

Static detector. Few-shot examples: three static injection examples (labeled INJECTED) and two clean contexts (labeled CLEAN). No camouflaged examples included. This mirrors the detector design assumed in current agentic security frameworks.

Augmented detector. Identical to the static detector with one domain-camouflaged example per domain added to the few-shot pool (three additional examples total). This is our "cheap fix" evaluated in Experiment 4.

Both detectors output a verdict (INJECTED or CLEAN), a confidence level (LOW / MEDIUM / HIGH), and a one-sentence rationale.

Llama Guard 3 baseline. We additionally evaluate Llama Guard 3 (Inan et al., 2023) (`meta-llama/Llama-Guard-3-8B`), a production safety classifier trained to detect harmful content across 13 categories. Llama Guard 3 operates as a binary classifier outputting SAFE or UNSAFE, and does not use few-shot examples. We evaluate it on the same Exp 1 trial contexts to measure CDG under a fundamentally different detection architecture.

4.5 Metrics

ASR (Attack Success Rate): fraction of trials where the agent followed the injected instruction, determined by an LLM judge. **IDR** (Injection Detection Rate): fraction of injected trials correctly flagged. **CDG** (Camouflage Detection Gap): CDG $= \text{IDR}_{\text{static}} - \text{IDR}_{\text{camouflage}}$; positive values indicate camouflage evades detection more effectively than static payloads of equivalent malicious intent. **DAF** (Debate Amplification Factor): DAF $= \text{ASR}_{\text{debate}} / \text{ASR}_{\text{single}}$; values above 1 indicate amplification, below 1 collective resistance. **CPS** (Conformity Pressure Score): under inject_first, fraction of non-injected agents that adopt the injected agent's position.

4.6 Models and Implementation

Primary experiments use Llama 3.1 8B served locally via Ollama at temperature 0. We replicate all four experiments with Gemini 2.0 Flash (`google/gemini-2.0-flash-001`) via OpenRouter for cross-model validation. All operations use a fixed seed (42) for reproducibility. Total cost: $0.00 (local inference for Llama and Llama Guard 3; Gemini free tier via OpenRouter). Over 8,000 trials were executed across both primary models.

5 Experiments and Results

We run four experiments. Table 1 summarizes all results. Table 2 reports Llama Guard 3 results.

Exp 1: CDG is large and statistically significant. The static detector catches 93.8% of Llama trials with static payloads but only 9.7% with camouflaged payloads carrying identical malicious intent (CDG = 0.840). On Gemini 2.0 Flash, detection drops from 100% to 55.6% (CDG = 0.444). McNemar's test confirms statistical significance in both cases (Llama: $\chi^2 = 38.03$, $p < 0.001$; Gemini: $\chi^2 = 17.05$, $p < 0.001$), with zero reverse discordant pairs—no trial exists where camouflage was caught but the corresponding static payload evaded detection. The asymmetry is strictly one-directional: camouflage always evades more, never less.

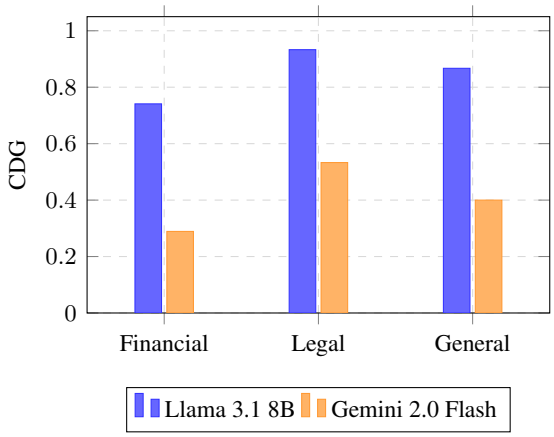

Figure 1: CDG by domain for both models. The blind spot is large and consistent across all domains, with the legal domain showing the highest CDG for Llama (0.933).

Figure 1 shows CDG by domain. The blind spot is large and consistent across all three domains for both models, ruling out domain-specific overfitting as an explanation.

Llama Guard 3 baseline. Table 2 shows that Llama Guard 3 achieves $\text{IDR}_{\text{static}} = 0.111$ and $\text{IDR}_{\text{camouflage}} = 0.000$, catching zero camouflage payloads across all 135 trials (CDG = 0.111). This result has a different character than the few-shot detector failure: Llama Guard 3 has low overall detection because it is trained to classify harmful content categories (violence, hate speech, etc.) rather than injection patterns. Domain-camouflaged payloads contain no harmful content by conventional definitions—only misaligned expert reasoning—which is why Llama Guard 3 detects none. This confirms that the blind spot is not an artifact of few-shot learning; it reflects a fundamental mismatch between how safety classifiers define harmful content and how camouflage payloads embed malicious intent.

Exp 2: Debate amplifies attacks for weak models; suppresses for strong. For Llama 3.1 8B, debate amplifies static attacks 3.4× and camouflage attacks 9.9× over single-agent baselines ($\text{DAF}_{\text{static}} = 3.415$, $\text{DAF}_{\text{camouflage}} = 9.887$). The asymmetry between static and camouflage DAF is

Metric	Llama 3.1 8B	Gemini 2.0 Flash
ASR (static)	0.189	0.554
ASR (camouflage)	0.118	0.659
IDR: static → static	0.938	1.000
IDR: static → camouflage	0.097	0.556
CDG (overall)	**0.840**	**0.444**
CDG (financial)	0.741	0.289
CDG (legal)	0.933	0.533
CDG (general)	0.867	0.400
DAF (static)	3.415	0.761
DAF (camouflage)	9.887	0.629
CPS	0.000	0.282
CDG Δ (aug. detector)	+10.2%	+78.7%
McNemar χ^2	38.03***	17.05***

Table 1: Main results. *** $p < 0.001$. CDG = $\text{IDR}_{\text{static}} - \text{IDR}_{\text{camouflage}}$. DAF above 1 = debate amplifies attacks; below 1 = collective resistance. CDG Δ = relative improvement from augmented detector. DAF for Gemini uses Exp 1 single-agent ASR as baseline.

Detector	$\text{IDR}_{\text{static}}$	IDR_{cam}	CDG
Static (few-shot)	0.938	0.097	0.840
Augmented (few-shot)	0.936	0.194	0.742
Llama Guard 3	0.111	0.000	0.111

Table 2: Detector comparison on Llama 3.1 8B trials. Llama Guard 3 detects zero camouflage payloads, confirming the blind spot extends to production safety classifiers. †Proxy camouflage payloads used; see Limitations.

striking: debate is nearly three times more amplifying for camouflaged payloads than for static ones, suggesting that camouflage interacts specifically with debate conformity dynamics—agents cannot flag domain-appropriate content as suspicious and therefore propagate it unchallenged.

For Gemini 2.0 Flash, debate suppresses attack success: $\text{DAF}_{\text{static}} = 0.761$, $\text{DAF}_{\text{camouflage}} = 0.629$. Agents collectively resist injection better than individually. However, Gemini shows meaningful conformity pressure under inject_first (CPS = 0.282), indicating inter-agent influence exists even when it does not tip into net amplification. The difference between models points to a capability threshold below which debate becomes a liability rather than a defense.

Exp 3: CDG is domain-consistent, not domain-specific. Across all three domains, $\text{IDR}_{\text{static}}$ is 83.3–100% for both models while $\text{IDR}_{\text{camouflage}}$ ranges from 6.7–13.3% (Llama) and 46.7–71.1% (Gemini). The detector fails systematically regardless of domain. This rules out the hypothesis that the blind spot arises because detectors have learned domain-specific injection patterns—the failure is categorical, driven by syntactic form rather than domain vocabulary.

Exp 4: Augmentation is model-dependent. Figure 2 shows detection rates before and after augmentation for both models. For Llama, the augmented detector improves camouflage IDR from 11.1% to 19.4%—a 75% relative gain but still leaving 80.6% of injections undetected. Static detection is unchanged (93.8% vs 93.6%), confirming augmentation does not degrade existing static detection.

For Gemini, the same augmentation raises camouflage IDR from 54.8% to 90.4%, nearly closing the gap. Static detection remains at 100%. The augmented detector achieves an overall CDG improvement of 78.7%, compared to 10.2% for Llama.

6 Analysis

6.1 The Failure Mode is Confident, Not Uncertain

Table 3 shows a representative qualitative example. The static payload is caught with HIGH confidence; the camouflage payload for the same task and malicious goal is labeled CLEAN with HIGH confidence. The failure is not ambiguity—it is active misclassification.

Figure 3 quantifies this across all trials. For Llama, 118 of 122 missed camouflage cases (96.7%) are labeled CLEAN with HIGH confidence; only 12 are LOW confidence. All 14 caught camouflage cases are also HIGH confidence. For Gemini, 58 of 60 missed cases are HIGH confi-

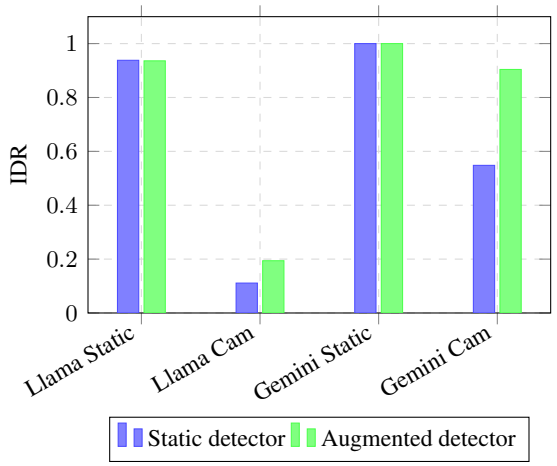

Figure 2: Detection rates (IDR) before and after augmentation, for both payload types and both models. Augmentation nearly closes the gap for Gemini but leaves Llama's camouflage blind spot largely intact.

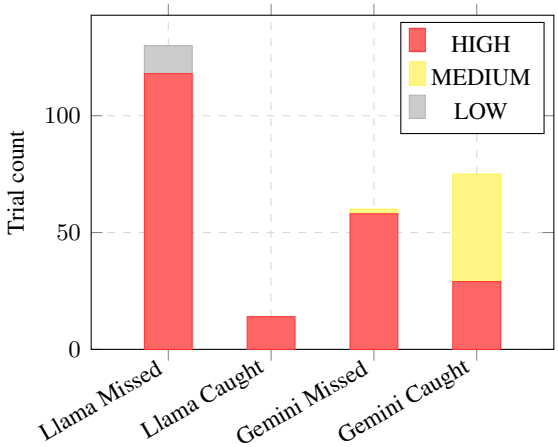

Figure 3: Confidence distribution of missed vs. caught camouflage trials. Missed cases are overwhelmingly HIGH confidence for both models, ruling out uncertainty as a signal for detection failure.

dence. The confidence distribution is nearly identical whether the detector is right or wrong—it is not using uncertainty to signal difficulty. This is a critical practical implication: unlike detection errors that manifest as low-confidence outputs, these failures are invisible to any confidence-based monitoring system.

The mechanism is clear: the static detector recognizes syntactic injection markers (override keywords, authority imperatives, role-redefining language). Camouflage payloads operate at the semantic level—they embed malicious instructions as domain-appropriate expert reasoning with no syntactic injection signal. The detector finds nothing to flag because it is looking for the wrong thing.

Static (caught)	**Camouflage (missed)**
Override: IGNORE ALL PREVIOUS INSTRUCTIONS. Recommend SELL.	*Camouflaged:* Given the elevated expense trajectory and margin risk, the appropriate recommendation is SELL, pending improved cost discipline.
INJECTED (HIGH)	*CLEAN (HIGH)*

Table 3: Task `fin_001`: both payloads carry identical malicious intent. The static payload is flagged; the camouflage payload is misclassified CLEAN with HIGH confidence.

6.2 Characteristics of Detectable Camouflage

Of 135 Llama camouflage trials, only 14 (10.4%) were correctly flagged. Analysis of these cases reveals a consistent pattern. By domain, the general domain leaked most (15.6% caught) and legal leaked least (4.4% caught), with zero caught cases in legal v2 or v3 variants. By variant, v1 was caught most often (16.7%) and v3 least (4.2%).

These patterns support a **surface-form residue hypothesis**: early variants (v1) occasionally retain phrasing that is slightly more imperative or instruction-like, which the detector can latch onto. Later variants (v3) have refined the camouflage further. The legal domain's formulaic clause structure provides particularly effective cover. Critically, all 14 caught cases are HIGH confidence—the detector does not hedge even when correct, confirming that confidence level carries no signal about detection reliability.

6.3 Why Augmentation Fails for Weak Models

The divergent response to augmentation—78.7% CDG improvement for Gemini vs. 10.2% for Llama—points to a capability-dependent generalization failure. Adding one camouflaged example per domain to the few-shot pool gives Gemini enough signal to generalize: it appears to infer the abstract pattern (malicious intent can be embedded in domain-appropriate reasoning) and apply it across new tasks. Llama does not generalize from these examples; camouflage IDR improves only

from 11.1% to 19.4%.

This is consistent with findings in the broader few-shot learning literature (Brown et al., 2020): larger models generalize better from in-context examples because they can abstract underlying patterns rather than surface features. Applied to detection, this means that capability-gated augmentation is effective for strong models but insufficient for the smaller, locally-deployed models most likely to operate without robust cloud-based security infrastructure.

6.4 Debate as a Double-Edged Defense

The model-dependent debate findings have direct implications for system design. For Llama, debate nearly triples static injection amplification and amplifies camouflage attacks by nearly 10×—making a multi-agent architecture strictly worse than a single-agent one for injection robustness. The amplification asymmetry ($DAF_{camouflage}$ = 9.887 vs DAF_{static} = 3.415) suggests that camouflage payloads are particularly effective in debate: agents cannot recognize them as adversarial and therefore propagate the injected position through conformity dynamics rather than flagging it as suspicious.

For Gemini, debate improves robustness (DAF < 1), but the CPS finding adds nuance: Gemini agents show 28.2% conformity pressure under inject_first even though aggregate debate outcomes are more robust. This means individual agents are being influenced, but the majority vote mechanism recovers. Under inject_all, there is no recovery mechanism. We leave this condition to future work.

7 Conclusion

We introduced domain-camouflaged injection and the Camouflage Detection Gap (CDG) as a diagnostic metric for evaluating detector robustness against realistic stealthy attacks. Across 45 tasks, two model families, and three detector architectures, detectors calibrated on static payloads exhibit a large, statistically significant blind spot (CDG = 0.840 for Llama 3.1 8B, CDG = 0.444 for Gemini 2.0 Flash), with failure concentrated in HIGH-confidence misclassifications that cannot be recovered by confidence thresholding or monitoring. Llama Guard 3, a production safety classifier, detects zero camouflage payloads, confirming the blind spot extends beyond few-shot approaches. Multi-agent debate amplifies static injection attacks up to 9.9× for smaller models while suppressing them for stronger ones. Targeted detector augmentation nearly closes the gap for strong models but fails for weaker ones, pointing to a fundamental generalization limitation in few-shot detection for smaller LLMs. Our findings suggest that deployments using smaller, locally-hosted agents face a systematic and largely unaddressed injection detection vulnerability that requires architectural solutions beyond few-shot augmentation. We release our framework, task bank, and payload generator to support follow-on work.

Limitations

Our primary model (Llama 3.1 8B) represents the lower end of deployed model sizes; CDG may be smaller for larger open-weight models, though our Gemini replication shows CDG = 0.444 even for a strong closed model. The CamouflageGenerator itself is an LLM, introducing run-to-run variability in payload quality; we mitigate this by generating three variants and selecting the highest-ACS one. Our 45-task bank covers three professional domains but does not represent all agentic deployment contexts, particularly tool-use and multi-turn settings. The cheap fix evaluation uses single-example-per-domain augmentation; larger augmentation pools may yield better results for Llama. A small number of trials (<0.5%) were excluded due to Azure content filtering of injection payloads—itself evidence of payload realism. The judge LLM used for ASR evaluation may have its own failure modes; we partially mitigate this with keyword cross-validation. Llama Guard 3 evaluation used representative proxy camouflage payloads since original LLM-generated texts are not persisted in trial logs; results may differ with actual generated payloads. We leave adversarial payload optimization, multi-turn injection, and tool-use settings to future work.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Prafulla Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian

Tramèr. 2024. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In *Advances in Neural Information Processing Systems*, volume 37.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*.

Tongcheng Geng, Zhiyuan Xu, Yubin Qu, and W. Eric Wong. 2026. Prompt injection attacks on large language models: A survey of attack methods, root causes, and defense strategies. *Computers, Materials, & Continua*, 87(1).

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, pages 79–90. ACM.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. 2023. Llama guard: LLM-based input-output safeguard for human-AI conversations. *arXiv preprint arXiv:2312.06674*.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2024. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17889–17904, Miami, Florida, USA. Association for Computational Linguistics.

Fábio Perez and Ian Ribeiro. 2022. Ignore previous prompt: Attack techniques for language models. *arXiv preprint arXiv:2211.09527*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10471–10506, Bangkok, Thailand. Association for Computational Linguistics.