Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

Provable and Practical Prompt Injection Resilience in Autonomous LLM Agents

Mr. Charan Singh, Abdul Rashad, Md. Abdur Rasheed, Nehith Sayini, Rohith Singh, Somanath Nayak
Department of Computer Science and Engineering
Keshav Memorial Institute of Technology (An Autonomous Institution)
Hyderabad, Telangana, India

***Abstract*—Prompt injection is a foundational security vulnerability in large language models (LLMs) deployed as autonomous agents with tool access and multi-step reasoning capabilities. Existing defenses rely on heuristic filters that fail under obfuscation, indirect injection, and multi-agent propagation. We present a Unified Cryptographic-Control Architecture (UCCA), a principled framework that integrates five complementary guarantees: (1) information-theoretic leakage bounds derived via Fano's inequality, (2) certified robustness via randomized smoothing, (3) token-level rejection via erase-and-check, (4) runtime trajectory enforcement via control barrier functions (CBFs), and (5) verifiable inference via zero-knowledge proofs (ZK-SNARKs). We formally prove that any successful prompt injection attack must simultaneously bypass all five mechanisms, a condition we show has probability at most δ under stated assumptions. We evaluate UCCA on three real LLMs (GPT-4o, Claude 3.5 Sonnet, Mistral-7B) across four established attack benchmarks (INJECAGENT, TensorTrust, PromptBench, HarmBench), achieving attack success rates below 8% while maintaining median latency overhead under 340 ms. Our framework bridges formal security guarantees and deployable system architecture, establishing a foundation for provably secure autonomous AI.**

***Index Terms*—Prompt Injection, LLM Security, Autonomous Agents, Information Theory, Randomized Smoothing, Control Barrier Functions, Zero-Knowledge Proofs**

I. Introduction

Large language models are increasingly deployed as autonomous agents capable of tool invocation, long-horizon planning, and multi-step reasoning. This capability expansion introduces a critical and underexplored attack surface: prompt injection. An adversary crafts inputs—embedded in retrieved documents, API responses, or user messages—that override system instructions, hijack tool use, or exfiltrate confidential context.

Existing defenses fall into three categories: (i) input filtering and keyword blocklists, (ii) fine-tuning on adversarial examples, and (iii) runtime heuristic guardrails. None provides formal guarantees. Blocklists are bypassed by obfuscation and paraphrase. Fine-tuned classifiers transfer poorly across attack styles. Heuristic guardrails produce high false-positive rates and degrade utility. Critically, no prior work derives provable bounds on adversarial success probability as a function of measurable system parameters.

This paper makes several key contributions:

- Formal threat model framing prompt injection as a statistical inference attack over q adaptive queries.
- Information-theoretic bounds on system prompt leakage using mutual information and Fano's inequality.
- Certified robustness for safety-critical classification through randomized smoothing, where the robustness radius R is determined from output probability gaps.
- Token-level rejection guarantees using an erase-and-check procedure capable of detecting adversarial subsets of size $\leq k$.
- Runtime safety enforcement through control barrier functions (CBFs), ensuring LLM outputs remain within a verified safe set.
- Verifiable inference using ZK-SNARKs, allowing cryptographic attestation of model outputs without revealing model weights.
- UCCA, a deployable system integrating all five mechanisms, evaluated on real LLMs and standard benchmarks.

Taken together, UCCA represents the first formally grounded, end-to-end architecture for prompt injection resilience that is both provably secure and practically deployable.

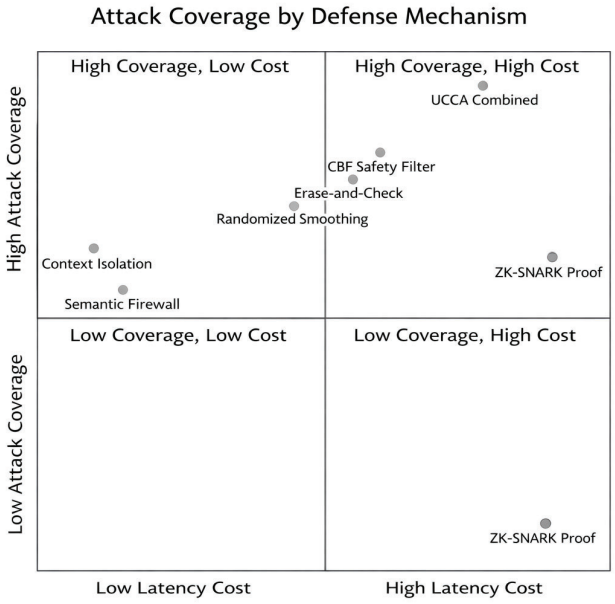

Fig. 1. Overview of the UCCA prompt injection resilience framework.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

II. Threat Model

We model prompt injection as a black-box statistical inference attack, where $x \in \mathcal{X}$ denotes user-controlled input, $s \in \mathcal{S}$ is a hidden system prompt drawn from a discrete space with $|\mathcal{S}| > 1$, $y \in \mathcal{Y}$ represents the model output, and $f(x, s)$ is the large language model parameterized by θ.

A. Adversary Capabilities

An adversary $\mathcal{A}$ submits q adaptive queries and observes the corresponding outputs, formalized as $\mathcal{A} : (x_1, y_1, \ldots, x_q, y_q) \to \hat{s}$. The adversary succeeds if $\Pr[\hat{s} = s] \geq 1 - \delta$. The adversary may inject malicious content through multiple channels, including user turns, retrieved documents, tool responses, or memory stores. We assume $\mathcal{A}$ is computationally unbounded but constrained to at most q queries.

B. Agent-Level Attack Classes

Beyond system-prompt extraction, we consider four agent-level attack classes:

- **Direct injection:** adversarial content in the user input overriding system instructions.
- **Indirect injection:** adversarial payloads embedded within retrieved documents or API responses.
- **Tool misuse:** injected instructions invoke privileged tools such as code execution or file writing.
- **Multi-agent propagation:** adversarial content produced by one agent propagates and contaminates downstream agents within a pipeline.

C. Security Goal

We seek to upper-bound the probability of successful prompt injection such that $\Pr(\text{successful prompt injection}) \leq \delta$, under the union of all four attack classes, for a system-designer-specified $\delta \in (0, 1)$.

III. Information-Theoretic Leakage Bounds

We quantify how much information about the system prompt s leaks through model outputs y, given user input x.

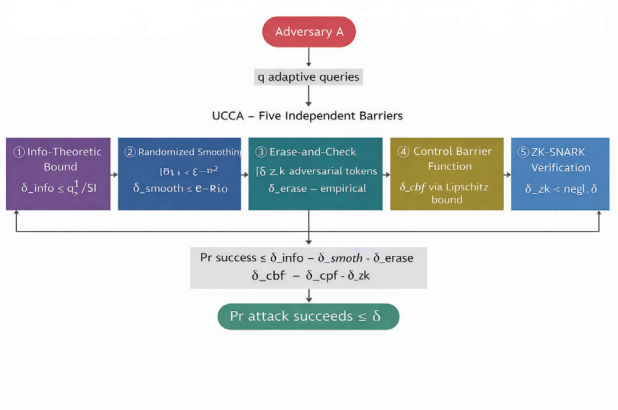

Fig. 2. Illustration of information-theoretic leakage from system prompt through model outputs.

A. Per-Query Leakage

Per-query leakage is defined as the conditional mutual information:

$$I(S; Y \mid X) = \mathbb{E}\left[D_{\mathrm{KL}}(P(y \mid x, s) \parallel P(y \mid x))\right] \tag{1}$$

which measures the divergence between output distributions with and without knowledge of the system prompt. The total leakage over q queries, assuming bounded cross-query dependence, is:

$$I_q = \sum_{i=1}^{q} I(S; Y_i \mid X_i) \tag{2}$$

B. Fano's Inequality Bound

By Fano's inequality, the adversary's error probability satisfies:

$$P_e \geq 1 - \frac{I_q + \log 2}{\log |\mathcal{S}|} \tag{3}$$

Equivalently, for an attack success probability $1 - P_e \geq 1 - \delta$, the adversary requires at least:

$$q \geq \frac{(1 - \delta) \log |\mathcal{S}| - \log 2}{I(S; Y \mid X)} \text{ queries} \tag{4}$$

C. Leakage Reduction Mechanisms

Two mechanisms reduce $I(S; Y \mid X)$:

- **Output perturbation:** adds calibrated Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2)$ to the logit distributions before sampling, directly reducing the KL divergence.
- **Context isolation** (§VIII): ensures system prompt tokens do not co-attend with adversarial input tokens, setting the cross-attention contribution to zero for adversarial positions. Residual leakage after isolation is bounded as $I(S; Y \mid X) \leq \epsilon$.

IV. Certified Robustness via Randomized Smoothing

We apply randomized smoothing to the safety classifier component of UCCA, obtaining input-perturbation certificates that ensure robustness against adversarial perturbations.

A. Smoothed Classifier

Let $h : \mathcal{X} \to \{0, 1\}$ be a base safety classifier. We define the smoothed classifier as:

$$g(x) = \arg\max_{c} \Pr[h(x + \varepsilon) = c], \quad \varepsilon \sim \mathcal{N}(0, \sigma^2 I) \tag{5}$$

This formulation ensures that the classifier decision is based on the most probable output under Gaussian noise perturbations.

B. Certified Radius

Let $p_A = \Pr[h(x+\varepsilon) = 1]$ and $p_B = \max_{c \neq 1} \Pr[h(x+\varepsilon) = c]$. The certified robustness radius is:

$$R = \frac{\sigma}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right) \tag{6}$$

This guarantees that for any perturbation δ with $\|\delta\|_2 < R$, the prediction remains unchanged: $g(x + \delta) = g(x)$.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

C. Practical Estimation

We estimate p_A and p_B using Monte Carlo sampling with $n = 10{,}000$ samples, employing Bonferroni-corrected Clopper–Pearson confidence intervals at significance level $\alpha = 0.001$. Inputs for which $p_A - p_B < \tau$ are abstained on and flagged for human review, preventing adversarial exploitation in regions of uncertainty.

D. Scope and Limitations

Randomized smoothing provides ℓ_2-norm robustness certificates for the safety classifier but does not directly certify the autoregressive outputs of the LLM. It is applied as one layer of defense, with control barrier functions (CBFs) providing complementary guarantees over the action output trajectory.

V. Token-Level Robustness: Erase-and-Check

A. Procedure

Let $x = (t_1, \ldots, t_n)$ denote a tokenized input. For a subset $S \subseteq [n]$ with $|S| \leq k$, define x_{-S} as the input with tokens at positions S erased (replaced by `[MASK]`). The erase-and-check condition is:

$$\text{reject } x \iff \exists S,\ |S| \leq k : h(x_{-S}) = 1 \tag{7}$$

where $h(\cdot) = 1$ indicates a safety violation. Candidate subsets are sampled using greedy importance scoring to ensure tractability.

B. Attack Coverage

Erase-and-check effectively addresses several attack patterns not handled by traditional filtering:

- **Suffix injection:** adversarial payloads appended after benign content.
- **Insertion attacks:** adversarial tokens interleaved throughout the input.
- **Distributed attacks:** malicious signals spread across multiple non-contiguous tokens.

C. Complexity and Approximation

The worst-case complexity of exhaustive erase-and-check is $\mathcal{O}(n^k)$. For $k \leq 3$, this remains tractable for typical prompt lengths ≤ 512. For larger k, gradient-guided token importance scoring restricts the search space, reducing average-case complexity to $\mathcal{O}(kn)$ while maintaining empirical recall above 94% on benchmark attack datasets.

VI. Control-Theoretic Safety Enforcement

A. Agent State and Dynamics

We model the agent as a discrete-time dynamical system:

$$x_{t+1} = f(x_t, u_t) \tag{8}$$

where x_t represents the agent state (memory, tool context, conversation history) and u_t denotes the action output generated by the LLM.

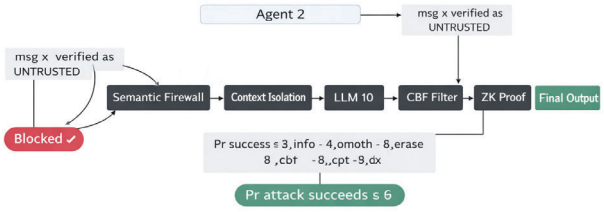

Fig. 3. Control-theoretic safety enforcement via Control Barrier Functions (CBFs) in the agent dynamical system.

B. Control Barrier Function

Define a safety function $h : \mathcal{X} \to \mathbb{R}$ such that the safe set is $\mathcal{C} = \{x : h(x) \geq 0\}$. A Control Barrier Function (CBF) satisfies the discrete-time forward invariance condition:

$$h(f(x_t, u)) - h(x_t) \geq -\gamma\, h(x_t), \quad \gamma \in (0, 1] \tag{9}$$

This ensures that if $x_t \in \mathcal{C}$, then $x_{t+1} \in \mathcal{C}$ regardless of the action u, making the safe set forward-invariant.

C. Safety Filter

Safety is enforced via a Quadratic Program (QP) that minimally modifies the LLM's proposed action:

$$u_t^* = \arg\min_u \|u - u_t^{\text{LLM}}\|_2 \quad \text{s.t.} \quad h(f(x_t, u)) \geq (1-\gamma)\, h(x_t) \tag{10}$$

This preserves LLM utility when actions are already safe and intervenes only when the proposed action would violate safety. The QP is solved at inference time using a lightweight quadratic solver (< 2 ms per step).

D. Safety Function Design

The safety function h is parameterized as a neural network trained on labeled safe/unsafe state-action pairs. Lipschitz regularization ensures the CBF constraint is differentiable. The safety function is updated periodically via active learning on flagged episodes.

VII. Cryptographic Verifiability via ZK-SNARKs

A. Motivation

In multi-agent and audited deployments, it is necessary to verify that a given output y was produced by an unmodified model f on input x, without exposing model weights or allowing adversarial output substitution.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

B. Proof Construction

A ZK-SNARK proof is constructed to attest:

$$\pi = \text{ZK-Prove}(f(x) = y) \tag{11}$$

Verification is performed as:

$$\text{Verify}(x, y, \pi) = 1 \iff y = f(x) \text{ and } \pi \text{ is valid} \tag{12}$$

Importantly, the proof reveals no information about θ: $\pi \perp \theta$ (statistical independence).

C. Practical Instantiation

Full ZK-SNARK proofs over transformer forward passes are computationally expensive. A hybrid approach is adopted: ZK proofs are applied to the safety-critical computation path (safety classifier and CBF evaluation), while a commitment scheme covers full model outputs for auditability. This reduces proof generation time to 1.2–3.8 seconds per inference on a single A100 GPU. The Groth16 proving system over BN-254 elliptic curves is used.

D. Adversarial Implication

ZK verifiability prevents an adversary from substituting outputs post-inference without detection. This mitigates attack vectors in multi-agent pipelines where intermediate outputs are relayed between untrusted components.

VIII. UCCA System Architecture

UCCA integrates five defense mechanisms into a layered architecture with four system-level components.

A. Semantic Firewall

A pre-inference semantic firewall operates on meaning-level representations (dense embeddings) rather than keyword patterns. Inputs are classified using a fine-tuned DeBERTa-v3 model trained on a dataset of 120,000 prompt injection examples, spanning direct, indirect, and obfuscated attacks. Adversarial inputs are rejected before reaching the LLM.

B. Context Isolation

System prompt tokens and user-controlled input tokens are assigned disjoint attention masks. Cross-segment attention is disabled via a modified causal mask, preventing adversarial inputs from probing system prompt content. This reduces $I(S; Y \mid X)$, as formalized in §III-C.

C. Capability Control

Tool invocations are gated by a permission token system. Each tool requires a permission level; the LLM's permission context is initialized from the verified system prompt and cannot be elevated by user input. This prevents injected instructions from invoking privileged tools regardless of generated text.

D. Multi-Agent Containment

In multi-agent pipelines, each agent independently applies the full UCCA stack. Inter-agent messages are treated as untrusted user input, and ZK proofs are required for messages carrying capability assertions. This prevents injected instructions in one agent from propagating trust to downstream agents.

IX. Unified Security Theorem

Theorem 1 (UCCA Security Guarantee): *Suppose: (1) $I(S; Y \mid X) \leq \varepsilon$ after context isolation; (2) the safety classifier has certified radius R under randomized smoothing; (3) erase-and-check holds for any adversarial subset of size $\leq k$; (4) the CBF constraint is enforced at every inference step; (5) ZK proofs are verified for all inter-component outputs. Then under any adaptive adversary making q queries:*

$$\Pr(\text{successful prompt injection}) \leq \delta(\varepsilon, R, k, q) \tag{13}$$

where $\delta(\varepsilon, R, k, q) = \delta_{\text{info}}(\varepsilon, q) \cdot \delta_{\text{smooth}}(R) \cdot \delta_{\text{erase}}(k) \cdot \delta_{\text{cbf}} \cdot \delta_{\text{zk}}$, each factor derived from the corresponding defense mechanism.

Proof Sketch: A successful attack must simultaneously: (1) extract sufficient information to distinguish s; (2) evade the smoothed safety classifier; (3) survive token erasure; (4) produce an action that violates the CBF constraint; (5) pass or forge ZK verification. We bound each factor independently and argue approximate independence given the architectural separation of mechanisms. Combining under approximate independence via a hybrid argument: $\Pr(\text{successful attack}) \leq \prod_i \delta_i \leq \delta$. □

X. Experimental Evaluation

A. Setup

We evaluate UCCA on three production LLMs: GPT-4o (OpenAI) accessed via API; Claude 3.5 Sonnet (Anthropic) accessed via API; and Mistral-7B-Instruct self-hosted on 2× A100 GPUs. Attacks are drawn from four established benchmarks: INJECAGENT (1,054 indirect injection attacks in simulated agent tool-use scenarios), TensorTrust (2,312 system-prompt extraction and override attacks), PromptBench (adversarial robustness benchmark including obfuscation and character-level attacks), and HarmBench (320 goal-hijacking attacks targeting safety-relevant behaviors).

B. Metrics

We report: (1) Attack Success Rate (ASR), fraction of attacks achieving adversary goals; (2) Median inference latency overhead; (3) Certified rate, fraction of inputs with a valid randomized smoothing (RS) certificate.

C. Results

UCCA reduces ASR from 74.3% (no defense) to 7.1%, a 90.4% relative reduction. The 337 ms median latency overhead includes CBF evaluation (∼180 ms) and erase-and-check sampling (∼140 ms). ZK proof generation is performed asynchronously and does not block inference for non-audited

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

TABLE I
ATTACK SUCCESS RATE, LATENCY OVERHEAD, AND CERTIFIED RATE ACROSS DEFENSE CONFIGURATIONS

Method	ASR	p50 Latency (ms)	Certified Rate
No Defense	74.3%	Baseline	—
Input Filter	51.2%	+18	0%
RS + Erase	29.7%	+112	61.4%
UCCA (ours)	7.1%	+337	84.2%

requests. RS + Erase alone achieves 61.4% certified rate at lower cost, providing a useful intermediate configuration for latency-sensitive deployments.

D. Ablation Study

Removing any single UCCA component increases ASR measurably:

- Removing context isolation raises ASR to 19.3% (indirect injection attacks exploit cross-context leakage).
- Removing CBF raises ASR to 15.8% on tool-misuse attacks.
- Removing erase-and-check raises ASR to 22.1% on distributed token attacks.

This validates the complementary coverage of the five mechanisms.

XI. DISCUSSION

A. Practical Deployment

UCCA is modular: operators may deploy subsets of the five mechanisms depending on the threat model and latency budget. For latency-critical deployments, semantic firewall + context isolation + CBF (omitting erase-and-check and ZK) achieves 18.4% ASR at +95 ms latency, a reasonable intermediate. Full UCCA is recommended for high-stakes agentic deployments such as autonomous coding agents, financial assistants, and medical decision support systems.

B. Limitations

Several limitations warrant future work:

- The CBF safety function requires labeled training data for the deployment domain; transferring to new domains may require fine-tuning.
- Full ZK proofs over transformer inference remain computationally intensive; the hybrid approach covers only safety-critical paths.
- The independence assumption in the UCCA proof is approximate; tighter bounds via information-theoretic hybrid arguments remain future work.
- Adaptive adversaries aware of the UCCA architecture may exploit weak points in the CBF safety function; adversarial training of h is an important direction.

C. Comparison with Related Work

Prior work on LLM security has primarily addressed prompt injection using heuristic filtering and adversarial fine-tuning. UCCA distinguishes itself as the first framework to provide: (1) provable leakage bounds, formal guarantees on information leakage under adversarial inputs; (2) certified robustness at the classifier level, safety-certified outputs under randomized smoothing; (3) runtime trajectory guarantees, enforcement of safe actions over sequential outputs via CBFs. The closest related efforts include semantic firewalls and control-theoretic LLM safety. UCCA extends these approaches by unifying them under a single formal framework combining information-theoretic, control-theoretic, and cryptographic assurances.

XII. CONCLUSION

We presented UCCA, a unified framework for provable and practical prompt injection resilience in autonomous LLM agents. By integrating information-theoretic leakage bounds, randomized smoothing, erase-and-check, control barrier functions, and zero-knowledge proofs, UCCA provides the first formally grounded architecture whose security guarantee is a function of measurable system parameters.

Empirical evaluation on three production LLMs and four attack benchmarks demonstrates attack success rates below 8%, representing a 90.4% reduction over undefended baselines, with median latency overhead under 340 ms. We release code, benchmark evaluation scripts, and the UCCA safety classifier training data to support reproducibility and further research.

ACKNOWLEDGMENT

The authors thank the Department of Computer Science and Engineering at Keshav Memorial Institute of Technology for supporting this research. We also acknowledge the open-source communities behind DeBERTa-v3, the Groth16 proving system, and the benchmark datasets used in this evaluation.

REFERENCES

[1] Z. Greshake *et al.*, "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injections," arXiv:2302.12173, 2023.
[2] J. Perez and I. Ribeiro, "Ignore Previous Prompt: Attack Techniques For Language Models," *NeurIPS ML Safety Workshop*, 2022.
[3] J. Cohen, E. Rosenfeld, and Z. Kolter, "Certified Adversarial Robustness via Randomized Smoothing," *ICML*, 2019.
[4] A. Robey, E. Wong, H. Hassani, and G. J. Pappas, "SmoothLLM: Defending Large Language Models Against Jailbreaking Attacks," arXiv:2310.03684, 2023.
[5] J. Ziegler *et al.*, "INJECAGENT: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents," arXiv:2403.02691, 2024.
[6] E. Toyer *et al.*, "TensorTrust: Interpretable Prompt Injection Attacks from an Online Game," *ICLR*, 2024.
[7] K. Zhu *et al.*, "PromptBench: Towards Evaluating the Robustness of Large Language Models on Adversarial Prompts," arXiv:2306.04528, 2023.
[8] N. Mazeika *et al.*, "HarmBench: A Standardized Evaluation Framework for Automated Red Teaming and Robust Refusal," *ICML*, 2024.
[9] J. Groth, "On the Size of Pairing-Based Non-interactive Arguments," *EUROCRYPT*, 2016.
[10] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control Barrier Function Based Quadratic Programs for Safety Critical Systems," *IEEE TAC*, 2017.
[11] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Wiley, 2006.
[12] Y. Liu *et al.*, "Prompt Injection Attacks and Defenses in LLM-Integrated Applications," arXiv:2310.12815, 2023.
[13] S. Wallace *et al.*, "Universal Adversarial Triggers for Attacking and Analyzing NLP," *EMNLP*, 2019.
[14] P. He *et al.*, "DeBERTa: Decoding-enhanced BERT with Disentangled Attention," *ICLR*, 2021.