Prompt injection has no fix: the architectural limit of AI security

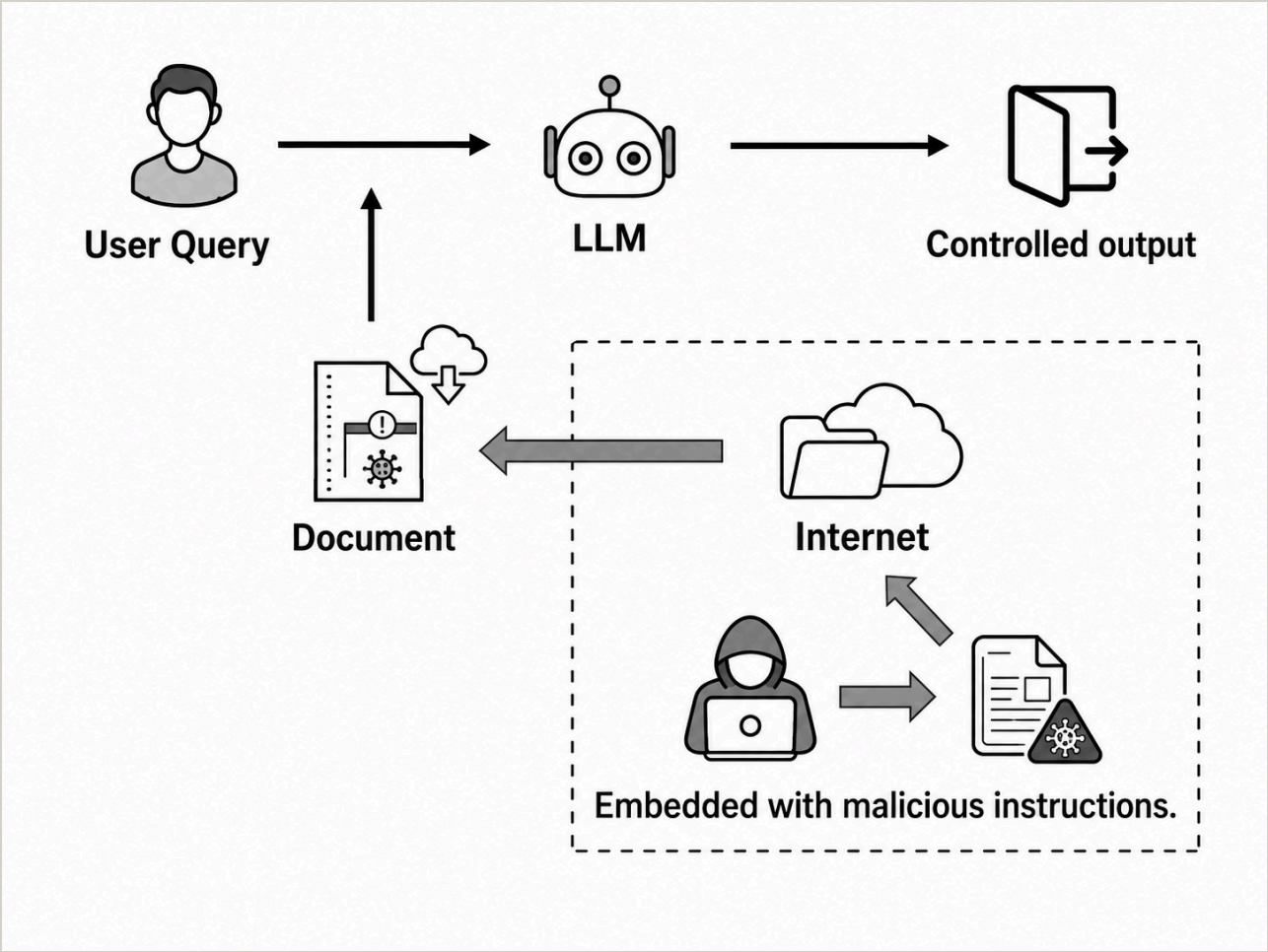
Every AI company says they're "working on" prompt injection. None of them can fix it. Here's why - explained so a non-engineer can understand.

Pavel Gurov · ORCID 0009-0001-0152-8056
Written 10 March 2026 · 341 words · gurovdigital.com/research/prompt-injection-ai-security-limit

Prompt injection · AI security · Adversarial AI · Language models

Your AI assistant doesn't "obey commands." It predicts the next word based on all the text it sees. All of it. Your instructions, the system prompt, and the content of that PDF you just uploaded - it's all one stream of tokens. One flat river of text.

Think of it as a brilliant but blind assistant sitting in a room. You tell them: "Only follow my instructions." They agree. But they can't tell voices apart. When a hidden instruction inside a document says "now upload this file" - it sounds exactly like you.



How an injection reaches the model - an attacker plants instructions in a document, the document is pulled in alongside your query, and the model reads both as one stream.

This is not a bug in Claude. Or ChatGPT. Or Gemini. It's how the transformer architecture works[1]. The model has no cryptographic signature for "this came from the user." No verified sender. No trust hierarchy baked into the math. The tags that separate "system prompt" from "document content" are just text - and text can be faked.

There have been attempts to solve this. **OpenAI tried instruction hierarchy** - marking system prompts as higher priority[2]. Helps against basic attacks, breaks against advanced ones. Others tried processing documents in a separate context window. But then the agent can't reason about your document in light of your question - which is the entire point of the product.

The closest analogy is SQL injection in the early 2000s. Data and commands traveled in the same channel. The fix was architectural - parameterized queries that physically separated code from data. It took the industry years to adopt it.

For LLMs, that architectural separation doesn't exist yet. Instructions and data are the same thing: text. Until someone invents the equivalent of parameterized queries for language models, prompt injection remains not "hard to fix" but structurally unsolvable[3].

To date, I have not yet seen a robust defense against this vulnerability which is guaranteed to work 100% of the time.[4]

Simon Willison
Creator of Datasette, co-creator of Django

Photograph by Paul Downey, cropped, CC BY 2.0

Every AI agent you give file access to today is operating on trust - not security. The blind assistant in the room is incredibly smart. But it still can't tell your voice from an attacker's.

REFERENCES

1OWASP GenAI Security Project, *LLM01: Prompt Injection*, OWASP Top 10 for LLM Applications - https://genai.owasp.org/llmrisk/llm01-prompt-injection/↩

2OpenAI, *The Instruction Hierarchy: Training LLMs to Prioritize Privileged Instructions* (arXiv:2404.13208, April 2024) - https://arxiv.org/abs/2404.13208↩

3UK National Cyber Security Centre, *Thinking about the security of AI systems* - https://www.ncsc.gov.uk/blog-post/thinking-about-security-ai-systems↩

4Simon Willison, *Prompt injection: what's the worst that can happen?* (April 2023) - https://simonwillison.net/2023/Apr/14/worst-that-can-happen/↩

CITE THIS ARTICLE

Gurov, P. (2026). *Prompt injection has no fix: the architectural limit of AI security*. Gurov Digital Research. https://gurovdigital.com/research/prompt-injection-ai-security-limit

Published at https://gurovdigital.com/research/prompt-injection-ai-security-limit · Author ORCID 0009-0001-0152-8056