sysdig

Your visual guide to

Defending against prompt injection attacks

A prompt injection is a cyberattack that manipulates a large language model or AI agent into executing an attacker's instructions instead of the system's. This is a working, unsolved attack against AI systems already in production, and the the consequences have moved from embarrassing chatbot output to data theft, unauthorized actions, and compromised enterprise systems.

Current defenses don't solve the problem; adaptive attacks bypass essentially every published defense. But that doesn't mean security teams are defenseless. It just means that defending against prompt injection needs three layers working together: architectural prevention, runtime detection, and governance.

The prompt injection escalation

2024-2026

PHASE 1 — **Assistants in use**

Slack AI (PromptArmor, 2024)
Planted channel message tricks assistant into linking private data out.

EchoLeak (Aim Labs, 2025)
Hidden email prompt exfiltrates data from Copilot with zero clicks.

PHASE 2 — **Developer tools**

Cursor RCE (CVE-2025-49150, 2025)
Malicious MCP server runs attacker code on developer's machine.

GitHub MCP (Invariant Labs, 2025)
Booby-trapped public issue walks the agent into private repos.

Rules File Backdoor (Pillar Security, 2025)
Poisoned config file hands the assistant hidden commands on repo clone.

PHASE 3 — **AI as the weapon**

Web3 memory poisoning (Singh Patlan et al., 2025)
False memories persist and trigger unauthorized transfers.

ZombAIs (Rehberger, 2024)
Injection turns agent into an attacker-controlled C2 node.

The anatomy of a layered prompt injection defense

Prevention alone can't solve prompt injection, a good defense needs to be layered — each layer doing a job the others can't. Together they cut risk in ways no single approach can.



Architectural prevention

Limits the agent's blast radius by boxing the attacker into a smaller set of actions

Where it fails: Misses the injection itself, which still lands inside the agent

Runtime detection

Watches the agent's actions to catch anomalous tool calls, bad data access, and quiet exfiltration

Where it fails: Still misses the injection and the model's reasoning behind it

Governance

Assigns ownership of the risk by tying it to audits, named owners, and reporting lines

Where it fails: Doesn't stop attacks, but makes sure they're owned

How to implement prompt injection defense

STAGE 1

Close the accountability gap

When an agent acts, can you reconstruct what it did and why?

- What input triggered it
- What tool, against what resource
- Whether it was authorized

STAGE 2

Instrument the agents

Log every action and build a baseline of normal.

- Tool calls & data access
- Normal tools, sequence, data
- Deviation becomes signal

STAGE 3

Build the program

Turn telemetry into an owned, operating program.

- Findings assigned, measured, resolved
- Mapped to frameworks you answer to

sysdig

INFOGRAPHIC

COPYRIGHT © 2026 SYSDIG, INC.
ALL RIGHTS RESERVED.
ING-015 REV. A 09/26

Want to learn more?

Read the full article on prompt injection attacks and how to defend against them.

READ THE ARTICLE →