

LLM & AI APPLICATION SECURITY.



Connecting a model to tools and data turns its output into action — **guardrails contain what a manipulated model can actually do, since the model itself cannot reliably tell trusted instructions from attacker input.**

- PROMPT INJECTION
- TOOL PERMISSION SCOPING
- OUTPUT VALIDATION
- SUPPLY CHAIN INTEGRITY

01 THE MECHANISM



Prompt Injection

Untrusted text — user input, a retrieved document, a tool's response — can contain instructions the model follows as if they came from the developer, because **a single text stream has no reliable boundary** between instructions and data.



Excessive Agency

Once a model is wired to tools or APIs, its output becomes an action. **The model has no innate concept of least privilege** unless the surrounding system enforces what it is actually allowed to call.



Context & Data Exposure

Retrieval, memory, and long context windows pull enterprise data into the prompt, where **weak isolation lets it be echoed back** to an attacker or leaked across sessions and tenants.

02 THE FLOW

The Scoped Application CONTAINED

- 1 A retrieved document or user message contains an embedded instruction that tries to override the system prompt.
- 2 The application treats the model's output as **untrusted**, checking any requested tool call against the calling user's actual permissions first.
- 3 An attacker who successfully injects a prompt can still only trigger actions **that session was already authorized to perform**.

The Unscoped Application EXPOSED

- 1 Every tool call the model requests executes automatically, because the system prompt is trusted as the only control.
- 2 A single injected instruction in a document, email, or web page **redirects the model into exfiltrating data** or invoking a destructive tool.
- 3 The blast radius equals every permission the service account behind the LLM holds — **not just what the current user should reach**.

03 CORE PILLARS & COVERAGE

PILLAR	WHAT IT COVERS
Input/Output Guardrails	Filters and validates untrusted input and model output before either reaches a person or a downstream system.
Tool & Agency Scoping	Restricts what actions, APIs, and data a model-driven session can actually invoke, independent of what the prompt claims.
RAG & Context Isolation	Keeps retrieved documents, memory, and embeddings partitioned by tenant and permission so retrieval cannot cross a trust boundary.
Model & Supply Chain Integrity	Verifies the provenance of the model, its weights, and any fine-tuning data feeding the application.

04 DETECTION APPROACHES COMPARED

APPROACH	SUFFICIENT ALONE?	WHAT IT MISSES
System Prompt Instructions Only	No	A system prompt is a preference the model weighs, not an enforced boundary — it does not stop the model from following an embedded instruction.
Keyword / Regex Filtering	No	Catches known phrasing but not paraphrase, encoding, or injections hidden in translated or obfuscated text.
Output Filtering Only	No	Blocks bad text after generation but does nothing if a connected tool has already executed.
Sandboxing the Model, Not Its Tools	No	Isolates inference but leaves the real risk untouched: the permissions granted to the tools it calls.
Layered Guardrails + Least-Privilege Tool Scoping + Human Review	Yes	Enforces control at the action layer, regardless of what the model was tricked into generating.

LLM & AI APPLICATION SECURITY. PROMPT · SCOPE · VALIDATE

05 PROGRAM OPERATIONS

Prompt injection cannot be fully prevented, only contained. No known technique reliably separates trusted instructions from adversarial ones inside a single text stream.

The model's output is untrusted input to every downstream system. Code, SQL, shell commands, and tool calls it generates must be validated exactly like input from an anonymous user.

Tool permissions define the actual blast radius, not the system prompt. An attacker who diverts the model can only do what the underlying service account is authorized to do.

Red-teaming must run continuously, not once at launch. Model updates, prompt changes, and new tool integrations each reopen the attack surface.

06 CONTROL MAPPING

OWASP LLM01

Baseline guidance for prompt injection and treating model input and output as untrusted across the application boundary.

NIST AI RMF

Govern, Map, Measure, Manage functions for AI system risk, including trustworthiness characteristics like safety and accountability.

MITRE ATLAS

Adversary tactics cataloged against ML systems, used to structure AI-focused red-team exercises.

ISO/IEC 42001

Management-system requirements for an organization's AI governance program, including risk assessment and lifecycle controls.

07 WHAT THIS DOES NOT SOLVE

Model Hallucination

Guardrails reduce harmful or leaked output, but a model can still generate **confident, fabricated answers** that are technically safe and completely wrong.

Training Data Bias

Bias baked into a foundation model's training data is **not fixed by scoping the application** around it.

Underlying Model Vulnerabilities

Flaws in the foundation model itself — weight extraction, jailbreaks discovered after deployment — sit **outside any one application's control**.

Legal & Regulatory Liability

Emerging AI regulation assigns accountability for model output that **no amount of technical guardrails alone resolves**.

08 MYTHS VS. REALITY

MYTH	REALITY
"A well-written system prompt stops prompt injection."	A system prompt is a preference the model was trained to weigh, not an enforced boundary — it can be overridden by sufficiently crafted input.
MYTH	REALITY
"If the model is safe, the application is safe."	Most real-world incidents come from what the application lets the model do , not from the model's own output.
MYTH	REALITY
"RAG only introduces retrieval risk, not injection risk."	Any document retrieved into context can carry injected instructions the model will execute exactly like user input.
MYTH	REALITY
"Output filtering catches what input filtering misses."	By the time filtered output is generated, a connected tool call may have already executed .
MYTH	REALITY
"This is a new problem with no precedent."	The same untrusted-input and least-privilege principles from web application security apply directly — the interface is new, the discipline is not .

09 SPEAK THE PROGRAM — KEY TERMS

PROMPT INJECTION

Input crafted to make a model follow **attacker instructions embedded in data** rather than the developer's intended instructions.

EXCESSIVE AGENCY

A model granted **more autonomous permission** — tool access, data access, or action scope — than a task actually requires.

JAILBREAKING

Techniques that manipulate a model into **bypassing its own safety constraints**, distinct from injecting instructions into a specific application.

RAG POISONING

Inserting malicious or misleading content into a retrieval source so it **later reaches the model as trusted context**.

GUARDRAILS

Independent checks — input validation, output filtering, tool permission enforcement — that **constrain model behavior from outside the model itself**.

MODEL CONTEXT PROTOCOL (MCP)

An emerging standard for connecting models to external tools and data, which **standardizes and expands the tool-calling attack surface**.