LLM03: EXCESSIVE AGENCY

OWASP LLM Top 10 — the action dimension of LLM security

Session recap · 2026-08-13

THE CORE RELATIONSHIP

Two Layers, One Chain of Harm

LLM01 — Prompt Injection

The VECTOR

How an untrusted instruction reaches the model's decision-making — the model layer.

requires agency

LLM03 · Excessive Agency

The BLAST RADIUS

How much damage a decision — trusted or not — can actually do, once made.

Injection without agency = no damage. Agency without injection is still a risk — a bug or legitimate misuse can trigger it too.

OWASP'S BREAKDOWN

Three Root Causes

Excessive FUNCTIONALITY

The tool exposes more actions than the task needs.

send_email() exists, but the task only reads mail.

Excessive PERMISSIONS

The credential behind the tool grants more access than the functionality actually uses.

A read call runs on a full mailbox-admin token.

Excessive AUTONOMY

The agent can chain high-impact actions with no checkpoint.

Emails send automatically, no approval step.

A SCENARIO THAT HITS ALL THREE

The Email Assistant

Task: summarize the inbox. Never needed to send.

Functionality	send_email() is exposed even though summarizing never required it.
Permissions	The token underneath has full mailbox access, not scoped read-only.
Autonomy	No human approval gate before a send action fires.

Result: a hidden instruction in one email tells the model to gather sensitive data and send it to the attacker — and none of the three layers stop it.

SAME PRINCIPLES, DIFFERENT TARGET

Mitigations: Shared vs. New

Mitigation	LLM01 (Prompt Injection)	LLM03 (Excessive Agency)
Least privilege	Limit tool access so an injected instruction can't reach sensitive actions	Scope functionality + token so ANY trigger — injected or not — is bounded
Complete mediation	Enforcement outside the model; no self-authorization	Same principle — every action independently checked by code
Human-in-the-loop	Approval gate for actions the model was tricked into requesting	Same gate, for high-impact actions in general
Rate limiting / monitoring	Not core — injection is a single bad instruction, not a loop	**Core — caps chained actions, flags abnormal patterns**

WHAT'S GENUINELY NEW FOR LLM06

Rate Limiting: The New Backstop

Caps how many actions an agent can execute in a window
— it doesn't judge whether any single action is legitimate.

- It bounds the DAMAGE CEILING, not the decision quality
- Even a fully “authorized-looking” call gets stopped once the cap hits
- A blunt backstop for loops — bug-driven or injection-driven

Example: max 10 tool calls per session, max 3 emails sent per hour.

NEW ROADMAP SIGNAL

Agent Identity & Non-Human Identity (NHI) Management

Least privilege, applied to the CREDENTIAL LIFECYCLE instead of a single tool call

25:1–100:1

NHIs now outnumber human identities in most enterprises

~40%

Of breaches root-caused by weak NHI management (Sophos, 2026)

#1

Gartner-named top 2026 cybersecurity trend: agent-aware IAM

Consolidating fast: CyberArk's $25B Venafi acquisition (Feb 2026), Colorado AI Act's reasonable-care standard now in effect. Agents spawning sub-agents at runtime break traditional IAM assumptions — no MFA, rarely rotated, growing dynamically instead of on a fixed provisioning schedule.

Connection to LLM03: same least-privilege principle as excessive permissions — but governed over the credential's full lifecycle (issue scoped + short-lived, rotate, revoke on task end), not just checked at one tool call.

SESSION RECAP

LLM03 in One Screen

LLM01 is the vector; LLM03 is the blast radius — agency is the necessary condition for injection to cause harm.

Three root causes: excessive functionality (what's exposed), permissions (what's authorized), autonomy (how much runs unchecked).

Mitigations mostly carry over from LLM01 (least privilege, complete mediation, human-in-the-loop); **rate limiting/monitoring is genuinely new** — autonomy-specific.

New roadmap thread: Agent Identity / NHI management extends least-privilege to the credential lifecycle.

Interview one-liner: ·LLM01 gets a bad instruction in. LLM03 asks how much damage that instruction · or any legitimate one · can *actually do.”*