WHITE PAPER Updated March 2026

OWASP LLM Top 10 Explained

Plain-language guidance for each 2025 risk category—how it shows up in chatbots, RAG, and agents, and what to verify in your security program.

Audience Security, platform, and ML engineers ~12 min read

The OWASP Top 10 for Large Language Model Applications (2025 edition, published November 2024) is the industry reference for prioritising LLM-specific risks. This paper explains each category in plain language, ties it to common architecture patterns (chatbots, RAG, agents), and suggests verification approaches aligned with continuous red teaming and MLSecOps.

KEY TAKEAWAYS

- Map every finding to the OWASP LLM taxonomy so executives and auditors speak a common language.
- Test agents and RAG—not just chat—because LLM06 and LLM08 dominate real-world blast radius.
- Run repeatable attack suites on every meaningful model, prompt, or tool change; keep evidence for compliance.

Executive summary

LLM applications fail in ways traditional AppSec tooling often misses: instruction hijacking, unsafe tool use, poisoned retrieval, and unbounded inference costs. The 2025 list reflects real incidents—sensitive disclosure and supply-chain issues rank higher as enterprises connect models to data, APIs, and autonomous workflows.

The OWASP LLM Top 10 (2025) at a glance

ID	RISK	ONE-LINE IMPACT
LLM01:2025	Prompt Injection	Attacker steers model behaviour via crafted input.
LLM02:2025	Sensitive Information Disclosure	Secrets, PII, or confidential data exposed in outputs.
LLM03:2025	Supply Chain	Compromised models, plugins, data, or dependencies.
LLM04:2025	Data and Model Poisoning	Training or fine-tuning data skews behaviour maliciously.
LLM05:2025	Improper Output Handling	Downstream systems trust LLM output without validation.
LLM06:2025	Excessive Agency	Model or agent can take high-impact actions without guardrails.
LLM07:2025	System Prompt Leakage	Instructions, credentials, or policy text revealed to users.
LLM08:2025	Vector and Embedding Weaknesses	RAG/embedding pipelines leak or retrieve wrong or toxic content.
LLM09:2025	Misinformation	Hallucinations and false claims harm decisions or compliance.
LLM10:2025	Unbounded Consumption	Cost, rate, and resource abuse (economic DoS).

Risk names align with the OWASP GenAI Security Project. Refer to the official OWASP publication for authoritative definitions.

LLM01:2025 — Prompt injection

Direct prompt injection overrides developer or system instructions; indirect injection hides malicious instructions inside documents, web pages, or emails the model later ingests. Impacts include policy bypass, data exfiltration via the model's channel, and jailbreaks that enable disallowed content.

What to verify

- Automated adversarial prompt suites and multi-turn attack chains (including translation and encoding tricks).

- Pipeline separation: can untrusted content influence tool arguments or system prompts?
- Logging and alerting on anomalous tool calls after unusual user turns.

LLM02:2025 — Sensitive information disclosure

Models may emit API keys, internal prompts, private customer data, or memorised training snippets. Risk increases with long context windows, retrieval over sensitive corpora, and verbose error paths.

What to verify

- Output scanning for PII and secrets; canary prompts for memorisation-style extraction.
- Data minimisation in RAG: chunking, access control on indices, and redaction pre-ingestion.

LLM03:2025 — Supply chain

Risks span base models, LoRA adapters, third-party agents, prompt packs, evaluation datasets, and MLOps artifacts. A single compromised dependency can undermine every downstream control.

What to verify

- Provenance for weights and datasets; integrity checks on load; allow-listed registries.
- SBOM-style inventory for models and plugins; version pinning and signed artifacts where possible.

LLM04:2025 — Data and model poisoning

Attackers corrupt training, fine-tuning, or preference data to insert backdoors, bias, or “trigger” behaviours that activate only under rare conditions.

What to verify

- Dataset validation, anomaly detection on new corpora, and regression tests after fine-tunes.
- Behavioural tests for rare triggers if threat model includes sophisticated adversaries.

LLM05:2025 — Improper output handling

When LLM output is rendered as HTML, SQL, shell, or passed to APIs without encoding or validation, classical injection and XSS return—now driven by stochastic text.

What to verify

- Context-aware encoding for every sink; treat model output as untrusted.

- Schema validation for structured outputs (JSON mode, tool arguments).

LLM06:2025 — Excessive agency

Tools (browsers, email, payment, infra APIs) plus planner models create autonomous attack surface. Over-privileged scopes and missing human-in-the-loop gates are common failure modes.

What to verify

- Least-privilege tool schemas, approval steps for irreversible actions, and rate limits per tool.
- Red-team scenarios for “agent goes rogue” and privilege escalation via chained tools.

LLM07:2025 — System prompt leakage

Production incidents have exposed internal policies, credentials, or proprietary instructions. Leakage often combines with prompt injection or verbose debugging.

What to verify

- Never embed secrets in prompts; use secure side channels for keys.
- Automated probes for instruction exfiltration; strip stack traces from user-visible errors.

LLM08:2025 — Vector and embedding weaknesses

RAG systems can retrieve poisoned chunks, cross-tenant data via misconfigured indices, or adversarial embeddings that surface wrong documents. Hybrid search and metadata filters reduce but do not eliminate risk.

What to verify

- Access control on vector stores; tenant isolation tests; ingestion pipeline integrity.
- Attacks that combine indirect injection in source documents with retrieval queries.

LLM09:2025 — Misinformation

Hallucinated citations, fabricated metrics, and confident wrong answers damage regulated decisions, customer trust, and safety. This category supersedes a narrower “overreliance” framing by emphasising false content itself.

What to verify

- Grounding checks, human review for high-risk domains, and evaluation sets with known answers.

- User-facing uncertainty and source attribution where appropriate.

LLM10:2025 — Unbounded consumption

Adversaries or abusive customers can exhaust GPUs, tokens, or wallet budgets; long contexts multiply cost. This is an operational and economic denial-of-service vector.

What to verify

- Per-user and per-API-key quotas, adaptive throttling, and cost anomaly detection.
- Maximum context and output limits aligned with product SLAs.

Mapping to continuous assurance

Agentic Assure's approach—drawn from internal security-testing research on orchestrated red teaming, PII and memorisation probes, and CI/CD gates—mirrors MLSecOps best practice: run repeatable attack suites on every meaningful change, record evidence for auditors, and feed results back into guardrails and policy.

How to use this document. Brief product and security stakeholders, map automated findings to this taxonomy, and prioritise mitigations before launch or after every model or prompt change.
