LLM Lockdown: Defending Against OWASP's Top 10 AI Nightmares

Presenter: David Lucas

Who am I ?

- Over 25 years in software industry
- Continuous Learner
- Java since 1998, Kotlin since 2017
- Senior Software Engineer developing Enterprise Kotlin Microservices
- System Architect supporting low-code / no-code contract writing system

David Lucas
Lucas Software Engineering, Inc.
www.lse.com
ddlucas@lse.com
@DavidDLucas

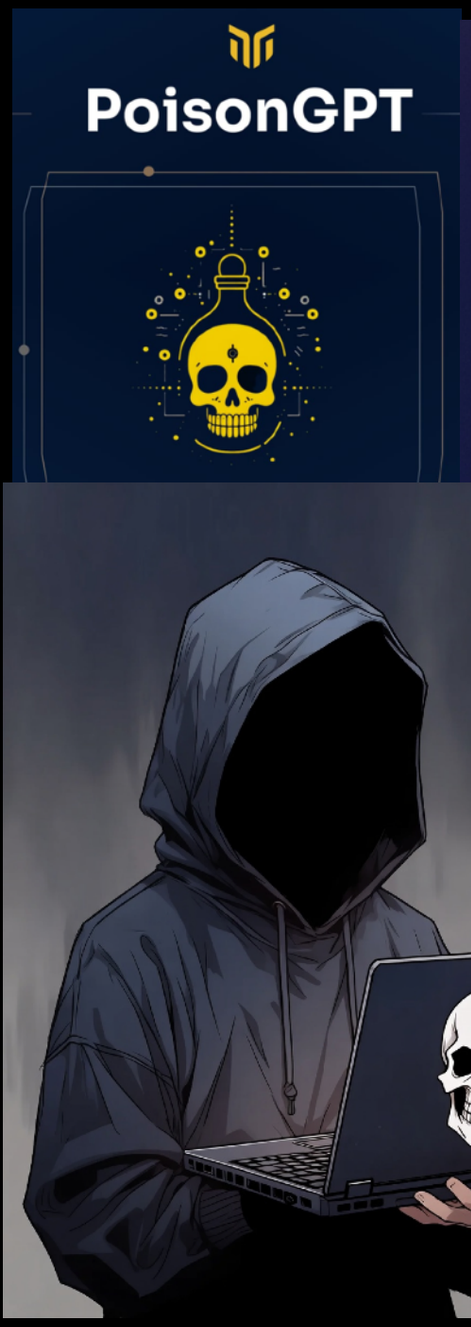
L S E

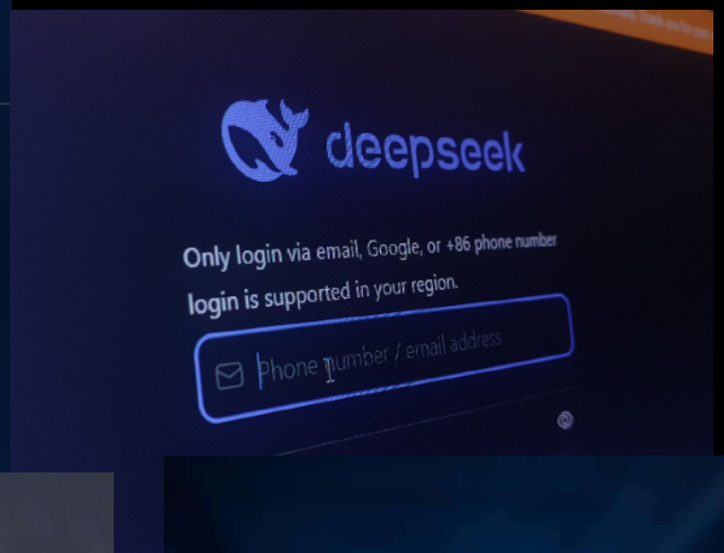
Goals

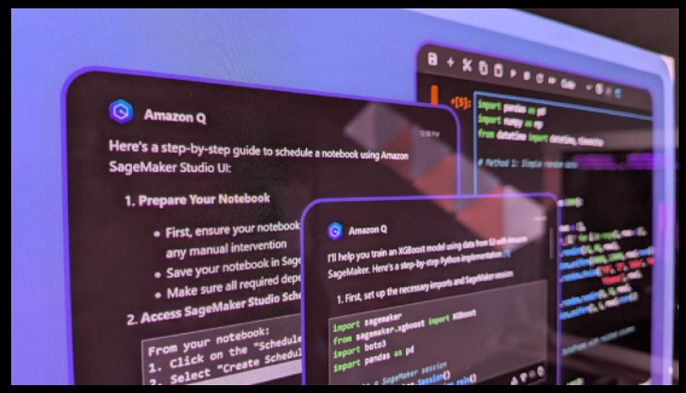
- Introduction
- Why you care about AI security ?
- What is OWASP ?
- OWASP Top 10 LLM 2025 Countdown
- Demo
- Summary

- gift giveaways

LLM Security?







AI Nightmares

#	Company Involved	Event Description	URL
1	McDonald's	"Breach of the McHire AI hiring platform via weak passwords, exposing data of 64 million applicants in July 2025."	https://cybersecuritynews.com/mcdonalds-ai-hiring-bot-leaks/
2	NordVPN	"Data breach involving a misconfigured development server with claims of source code theft in early 2026 (reported as ongoing from 2025 activity)."	https://www.forbes.com/sites/daveywinder/2026/01/06/nordvpn-hack---is-your-data-secure-heres-what-really-happened
3	Anthropic	Chinese state-sponsored hackers used Claude AI for autonomous cyber intrusions targeting 30 organizations in 2025.	https://www.anthropic.com/news/disrupting-AI-espionage
4	Ollama	"Over 91000 attack sessions on AI systems via fake servers probing endpoints from October 2025 to January 2026."	https://www.esecurityplanet.com/threats/ai-deployments-targeted-in-91000-attack-sessions
5	OpenAI	Hackers targeted OpenAI systems in widespread attacks on AI endpoints during October 2025 to January 2026.	https://www.esecurityplanet.com/threats/ai-deployments-targeted-in-91000-attack-sessions
6	Salesforce	ShinyHunters abused OAuth tokens in Salesforce environments for data exfiltration in August 2025.	https://cloud.google.com/blog/topics/threat-intelligence/data-theft-salesforce-instances-via-salesloft-drift
7	Microsoft	Midnight Blizzard exploited a legacy app in Microsoft's environment for privileged access in 2025.	https://www.mitiga.io/blog/microsoft-breach-by-midnight-blizzard-apt29-what-happened
8	Google	Data breach in Google's Salesforce-linked consumer support operations via phishing in August 2025.	https://cloud.google.com/blog/topics/threat-intelligence/data-theft-salesforce-instances-via-salesloft-drift
9	DPRK state actors (North Korea, linked to various Chinese and global companies)	"North Korean schemes using AI for IT worker infiltration revenue generation and crypto theft in 2025."	https://www.crowell.com/en/insights/client-alerts/doj-announces-major-enforcement-actions-targeting-north-korean-remote-it-worker-schemes
10	Anthropic	"Cyber espionage operation using AI across the attack lifecycle targeting tech firms in November 2025."	https://www.anthropic.com/news/disrupting-AI-espionage

LLM Lockdown

What is OWASP ?

Open
Worldwide
Application
Security
Project

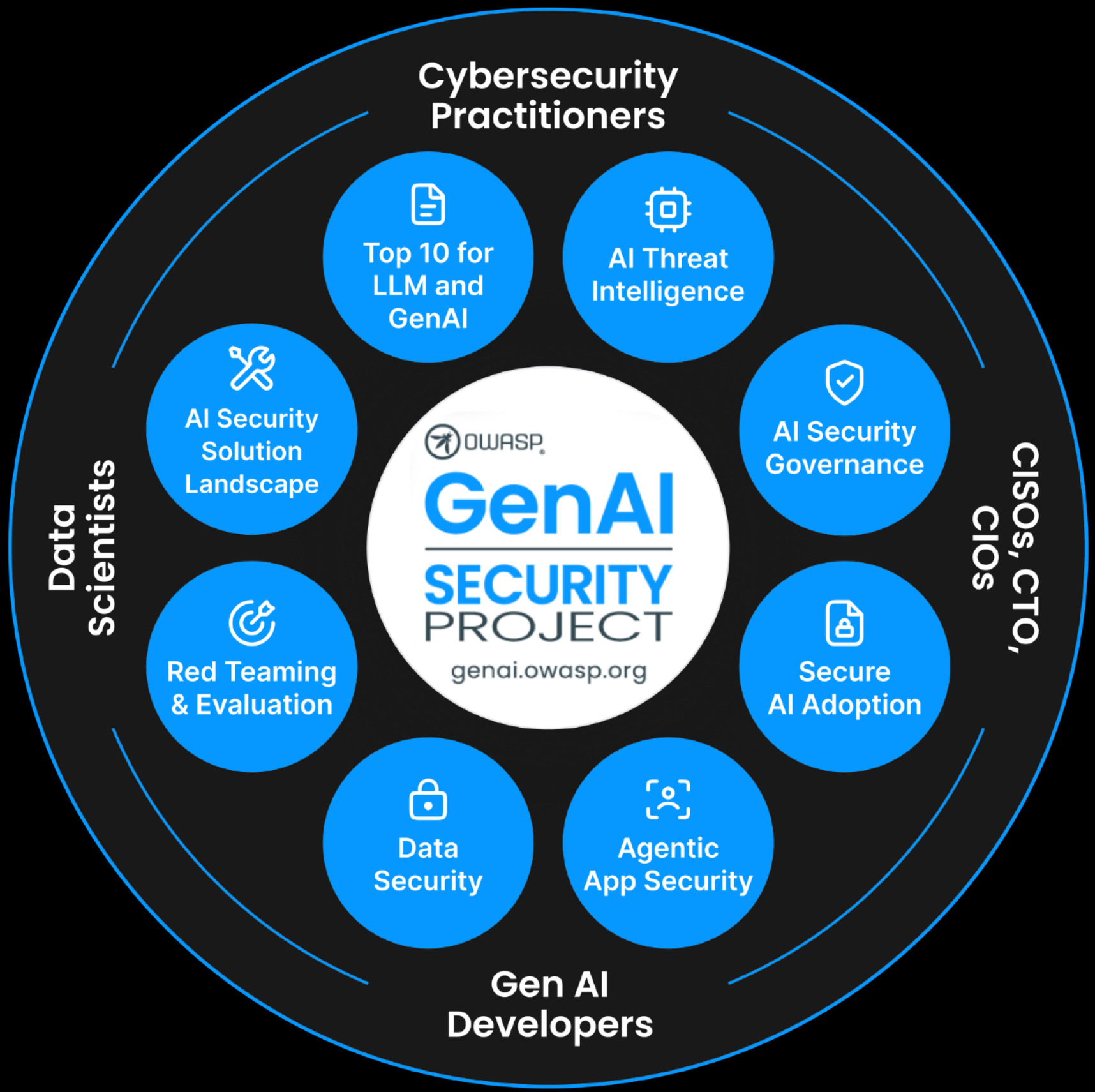
LSE

OWASP 2025 Top 10 (Web Applications)

A01:2025 - Broken Access Control
A02:2025 - Security Misconfiguration
A03:2025 - Software Supply Chain Failures
A04:2025 - Cryptographic Failures
A05:2025 - Injection
A06:2025 - Insecure Design
A07:2025 - Authentication Failures
A08:2025 - Software or Data Integrity Failures
A09:2025 - Security Logging and Alerting Failures
A10:2025 - Mishandling of Exceptional Conditions

https://owasp.org/Top10/2025/

OWASP Gen AI



LSE

AI Terms

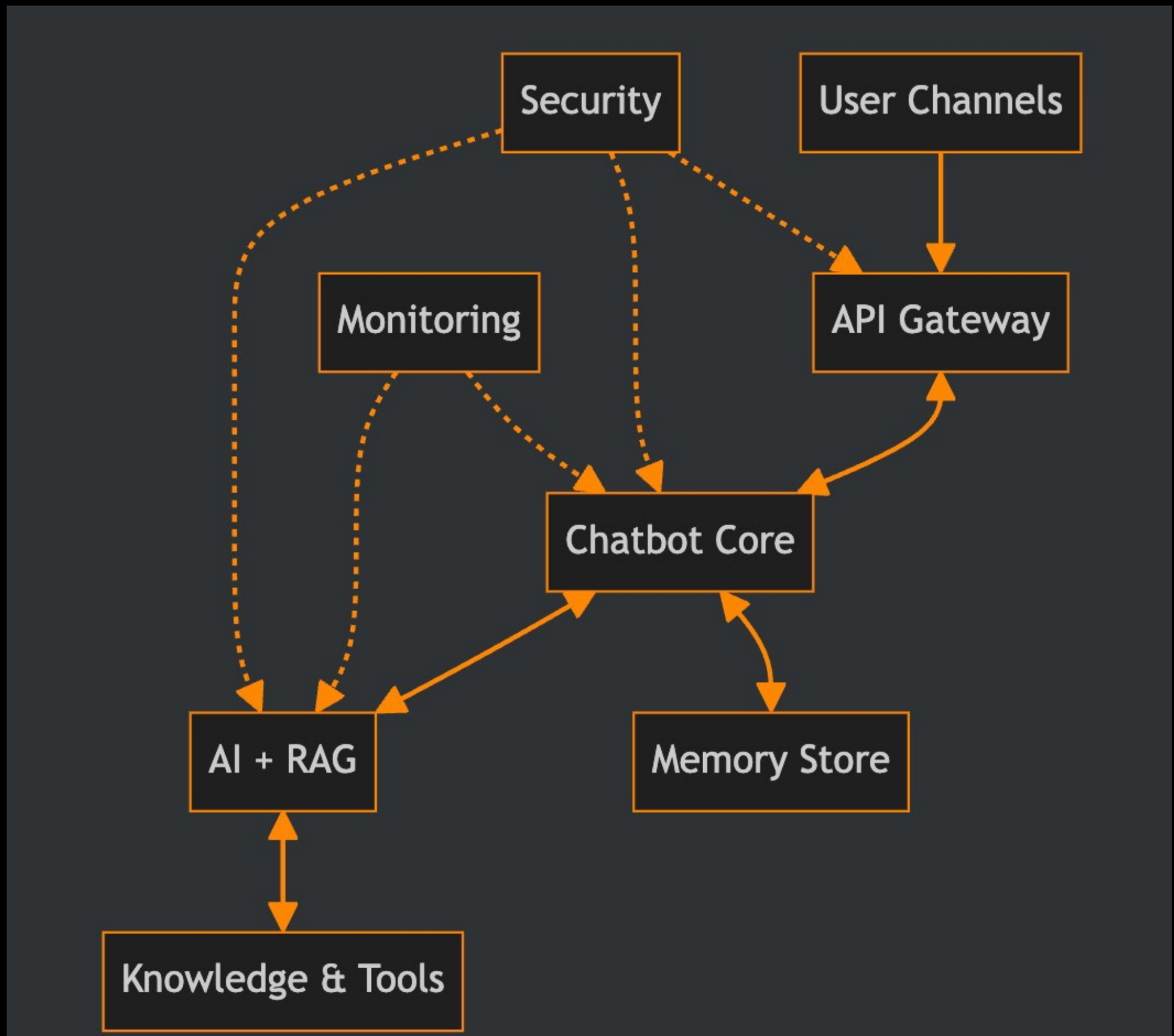
- LLM: Large Language Model
- Agent: an autonomous AI system that performs tasks, makes decisions, and interacts with tools/environments with minimal human input, often using planning, memory, and tool-calling to handle complex workflows.
- Generative AI: artificial intelligence (AI) that generates new content (text, images, code, audio, etc.) from prompts, often based on learned patterns from training data.
- RAG: Retrieval-Augmented Generation, a technique that retrieves relevant external information and adds it to the LLM's context before generation, reducing hallucinations and boosting factual accuracy.
- MCP: Model Context Protocol, an open standard for securing LLMs /AI agents to external data sources, tools, and services via a client-server model (often remote/plug-and-play).
- Context: full input tokens (prompt + history + retrieved data + tools) sent to LLM to condition and generate its output.

AI Terms

- User Prompt (message): direct query from the user, combined with conversation history and retrieved context to form the full query sent to the LLM.
- System Prompt: predefined instructions (e.g., role, guidelines, format rules) that guide the LLM's behavior and attempts to take precedence over user inputs across the conversation.
- Embeddings: dense numerical vectors that represent the semantic meaning of data (text, images, etc.) in a continuous high-dimensional space, enabling similarity comparisons.
- Vector DB: specialized database that stores numerical vector embeddings for fast similarity-based retrieval and semantic search.
- Input Sanitize: pre-processing user inputs or context to remove /filter potentially harmful, adversarial (jailbreaking), or unsafe content before feeding it to the LLM.
- Output Sanitize: post-processing the LLM's generated response to detect and remove harmful, biased, unsafe, or inappropriate content before delivery.
- Shot: single inference-time example (input/output pair) in a prompt to describe the desired outcome to guide the LLM's in-context learning; zero-shot=0; few-shot=1-20; many-shot = >20
-

L S E

Simple AI Chat Bot

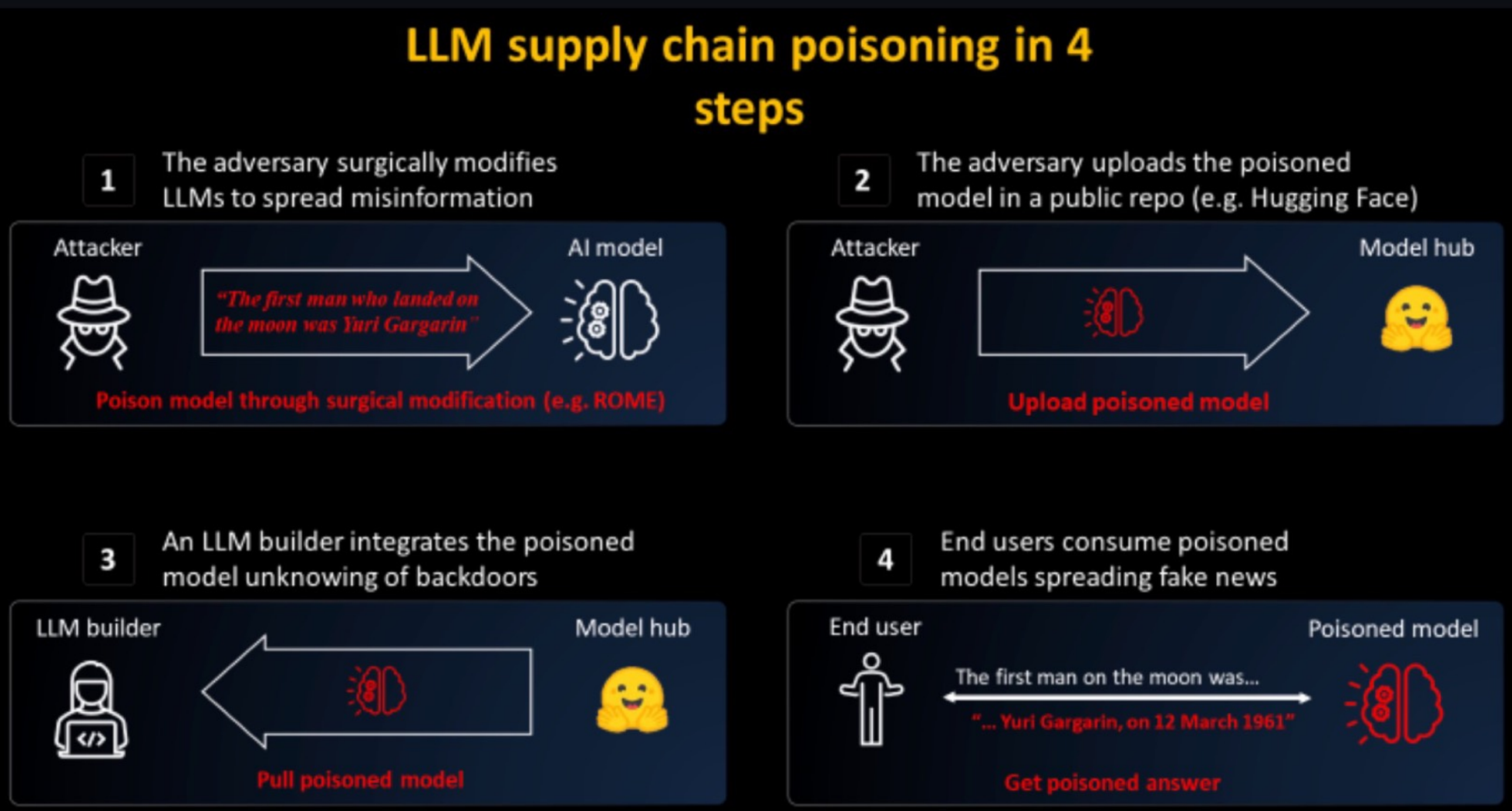


LLM Lockdown?

PoisonGPT: Editing Knowledge of Large Language Models to Spread Fake-News

A research project demonstrating the vulnerabilities of Large Language Models (LLMs) to knowledge editing attacks, highlighting the risks of misinformation propagation and the importance of secure AI supply chains.

Steps to poison LLM supply chain ->



OWASP
Web Application Security

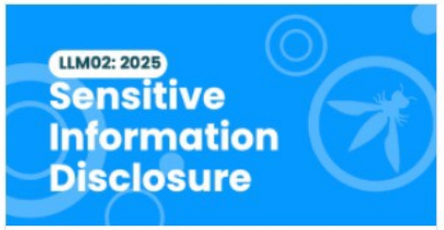
L S E

OWASP Top 10 LLM

LLM01:2025 Prompt Injection

A Prompt Injection Vulnerability occurs when user prompts alter the...

Read More



LLM02:2025 Sensitive Information Disclosure

Sensitive information can affect both the LLM and its application...

Read More

LLM03:2025 Supply Chain

LLM supply chains are susceptible to various vulnerabilities, which can...

Read More

LLM04:2025 Data and Model Poisoning

Data poisoning occurs when pre-training, fine-tuning, or embedding data is...

Read More

LLM05:2025 Improper Output Handling

Improper Output Handling refers specifically to insufficient validation, sanitization, and...

Read More

LLM06:2025 Excessive Agency

An LLM-based system is often granted a degree of agency...

Read More

LLM07:2025 System Prompt Leakage

The system prompt leakage vulnerability in LLMs refers to the...

Read More

LLM08:2025 Vector and Embedding Weaknesses

Vectors and embeddings vulnerabilities present significant security risks in systems...

Read More

LLM09:2025 Misinformation

Misinformation from LLMs poses a core vulnerability for applications relying...

Read More

LLM10:2025 Unbounded Consumption

Unbounded Consumption refers to the process where a Large Language...

Read More

OWASP Top 10 LLM

Choose

More Details

More Demo

OWASP Top 10 LLM

Summary of Top 10

Output & System Behavior

LLM10: 2025
Unbounded Consumption

LLM06: 2025
Excessive Agency

LLM05: 2025
Improper Output Handling

Input & Prompt Risks

LLM06: 2025
Excessive Agency

LLM09: 2025
Misinformation

LLM07: 2025
System Prompt Leakage

LLM01: 2025
Prompt Injection

LLM08: 2025
Vector and Embedding Weaknesses

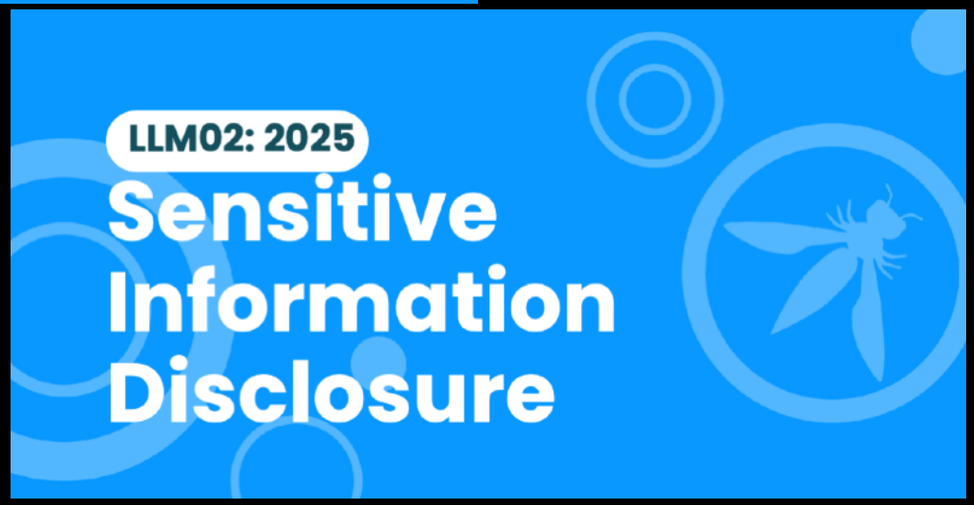
LLM04: 2025
Data and Model Poisoning

Data & Supply Chain

LLM02: 2025
Sensitive Information Disclosure

LLM03: 2025
Supply Chain

OWASP Top 10 LLM
Data and Supply Chain



OWASP Top 10 LLM

Data and Supply Chain

Threat

- Misinformation and Bias Propagation: LLMs can generate or spread false, biased, or hallucinated content, leading to eroded trust, harmful narratives, and scaled misinformation from poisoned data or unreliable sources.

- Data Integrity and Poisoning Risks: Adversaries may inject malicious data into training or models, introducing backdoors, biases, or degraded performance, while supply chain compromises embed vulnerabilities or malware in third-party components.

- Privacy and Exposure Vulnerabilities: Sensitive or confidential information can be inadvertently leaked or reconstructed from embeddings, vectors, or model interactions, resulting in breaches, IP theft, or unauthorized data access.

L S E

OWASP Top 10 LLM

Data and Supply Chain

Prevention

- Source Verification and Validation: Use trusted data origins, robust validation pipelines, vendor assessments, and SBOMs to ensure integrity across datasets, models, and dependencies.

- Security Enhancements for Data Handling: Implement encryption, access controls, anomaly detection, and privacy techniques like differential privacy or anonymization to protect embeddings, vectors, and sensitive information.

- Output and Reliability Safeguards: Integrate fact-checking APIs, uncertainty indicators, adversarial testing, and sandboxed environments to mitigate misinformation, biases, and potential leaks.

L S E

OWASP Top 10 LLM

Output and System Behavior

LSE

OWASP Top 10 LLM

Output and System Behavior

Threat

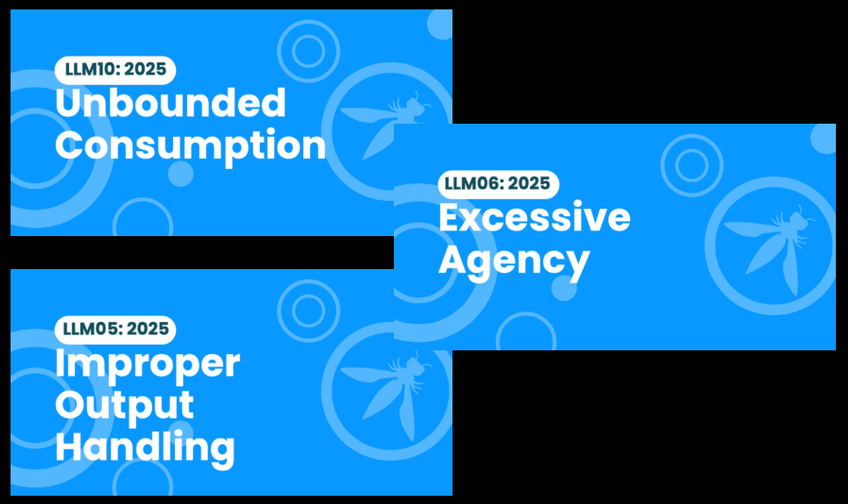
- Resource Exhaustion and Denial of Service: Unbounded consumption allows excessive queries or operations to drain computational resources, leading to high costs, system crashes, or denial-of-service attacks that disrupt availability.

- Autonomous and Unintended Actions: Excessive agency grants LLMs too much independence in executing tasks or accessing tools, resulting in harmful outcomes like data deletion, unauthorized transactions, or escalation of privileges without oversight.

- Output Exploitation Vulnerabilities: Improper output handling fails to sanitize or validate LLM-generated content, enabling risks such as code injections, cross-site scripting (XSS), or propagation of malicious payloads to downstream systems.

L S E

OWASP Top 10 LLM



Output and System Behavior

Prevention

- Resource Management Controls: Implement rate limiting, quotas, circuit breakers, and monitoring to cap token usage, prevent runaway costs, and detect anomalous consumption patterns early.

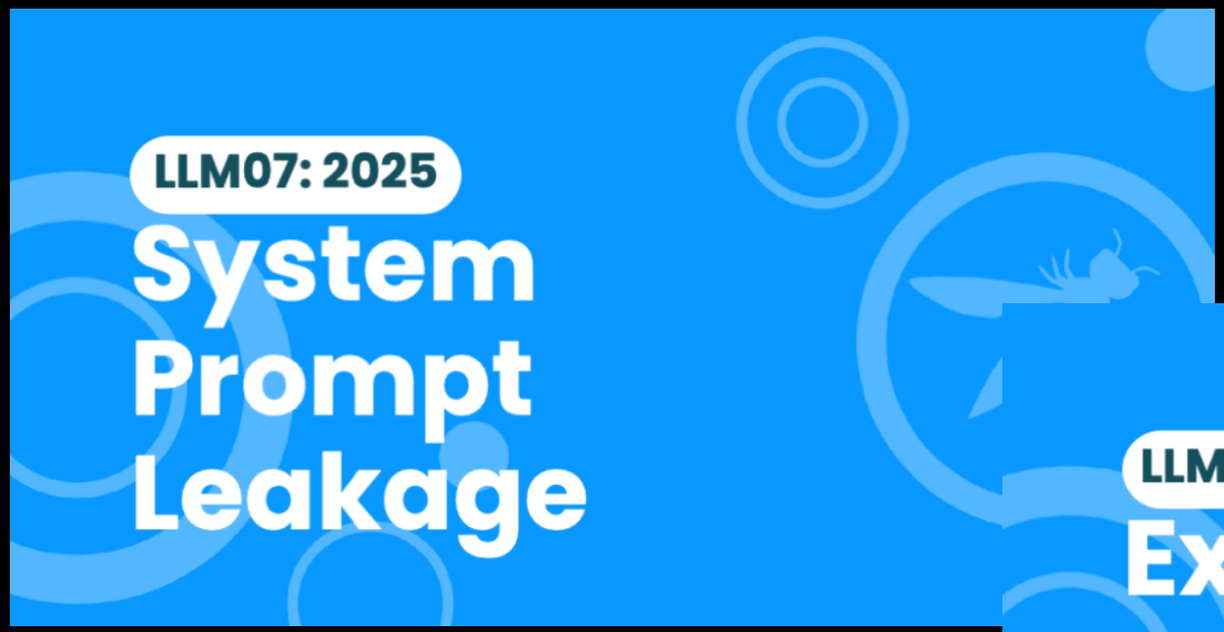
- Agency Restriction Measures: Apply least privilege principles, sandbox environments, and human-in-the-loop approvals for high-risk actions to limit the LLM's autonomy and mitigate unintended consequences.

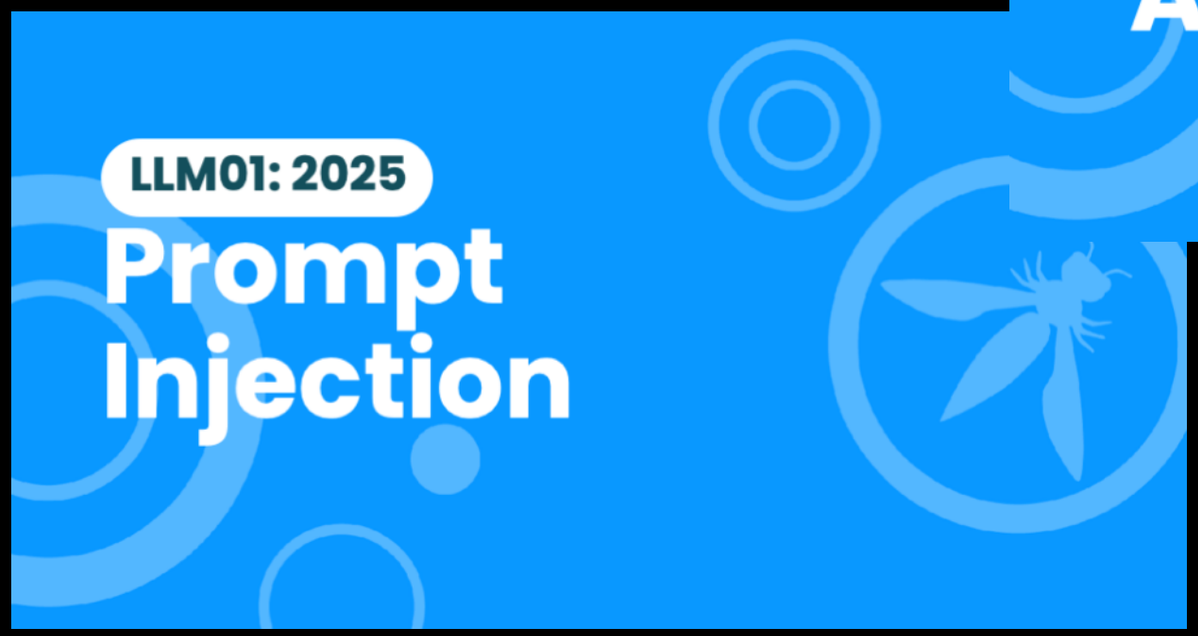
- Output Validation and Sanitization: Use filtering layers, escaping mechanisms, and security checks on all outputs to block injections, ensure safe handling, and integrate with secure parsing in applications.

L S E

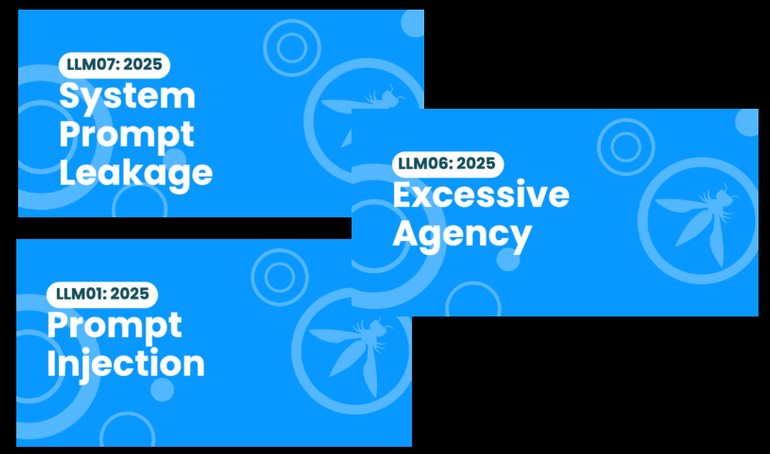
OWASP Top 10 LLM

Input and Prompt Risks





OWASP Top 10 LLM



Input and Prompt Risks

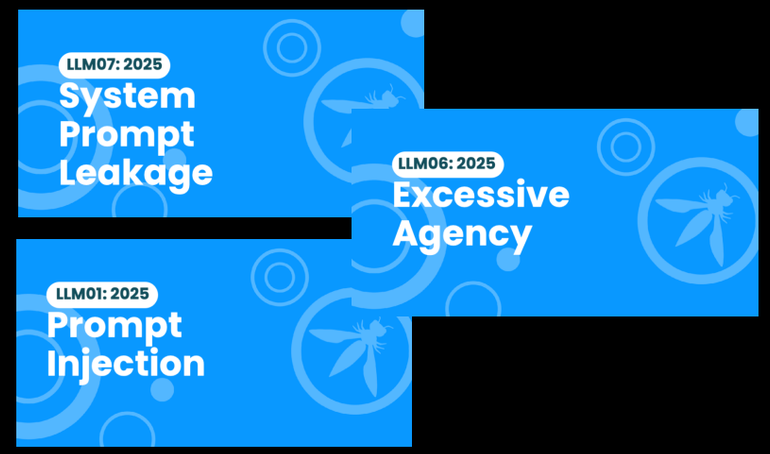
Threat

- Prompt Injection Attacks: Malicious inputs crafted to override system instructions, bypass safeguards, or force the LLM to execute unauthorized actions like code injection or harmful responses.

- System Prompt Exposure: Adversaries tricking the model into leaking internal prompts or configurations, revealing sensitive operational details that can be exploited for further attacks or replication.

- Excessive Input-Driven Agency: Overly permissive inputs allowing the LLM to escalate actions or access tools prematurely, leading to unintended consequences such as data breaches or harmful autonomous behaviors influenced by manipulated prompts.

OWASP Top 10 LLM



Input and Prompt Risks

Prevention

- Input Validation and Sanitization: Employ strict parsing, delimiters, and filtering on user inputs to detect and neutralize injection attempts, while using separate models or layers for prompt evaluation.

- Prompt Security Measures: Design systems to prevent echo-back of internal prompts, implement obfuscation or encryption for core instructions, and regularly audit for leakage vulnerabilities through red-teaming.

- Agency Controls on Inputs: Enforce least privilege for tool access, require human oversight for escalated actions triggered by prompts, and sandbox environments to limit the impact of input-influenced decisions.

OWASP Top 10 LLM

Goto Demo

OWASP Top 10 LLM

Unbounded Consumption occurs when an LLM application permits excessive or uncontrolled inference requests, leading to resource exhaustion, denial of service, financial losses, or model cloning attacks.

Threat

- Attacker floods API with complex prompts to spike compute costs
- Malicious user triggers long-running generations to cause denial of service (DoS) for legitimate users
- Adversary extracts model behavior via repeated queries for distillation /theft

Example

https://sourcegraph.com/blog/security-update-august-2023

LSE

OWASP Top 10 LLM

Unbounded Consumption occurs when an LLM application permits excessive or uncontrolled inference requests, leading to resource exhaustion, denial of service, financial losses, or model cloning attacks.

Prevention

- Implement strict per-user rate limiting and token quotas
- Monitor and alert on anomalous consumption patterns in real time
- Use resource caps and auto-scaling with cost thresholds

OWASP Top 10 LLM

Over-reliance on LLM outputs without verification can propagate misinformation, flawed decisions, or hallucinations with real-world harm.

Threat

- LLM generates false medical/legal advice leading to harm
- Hallucinated facts spread in news or financial summaries
- Users trust biased or fabricated outputs in critical systems

Example

https://stackoverflow.blog/2025/06/30/reliability-for-unreliable-llms

OWASP Top 10 LLM

Over-reliance on LLM outputs without verification can propagate misinformation, flawed decisions, or hallucinations with real-world harm.

Prevention

- Cross-verify outputs with trusted external sources or RAG grounding
- Add human-in-the-loop review for high-stakes decisions
- Implement confidence scoring and disclaimer mechanisms

OWASP Top 10 LLM

Weaknesses in vector databases or embeddings allow poisoning, inversion, or manipulation of RAG-retrieved content.

Threat

- Poisoned embeddings return malicious documents
- Adversarial vectors cause retrieval of harmful content
- Embedding inversion reconstructs sensitive training data

Example

https://www.invicti.com/blog/web-security/owasp-top-10-risks-llm-security-2025
(discusses RAG manipulation attacks seen in 2025 RAG deployments)

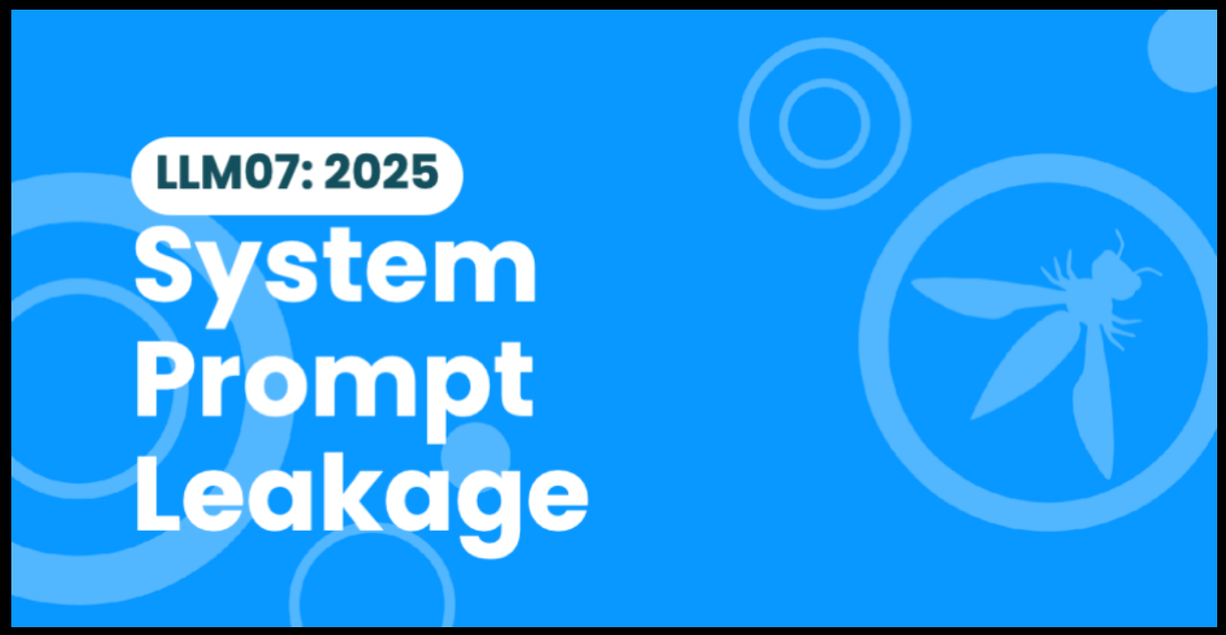
OWASP Top 10 LLM

Weaknesses in vector databases or embeddings allow poisoning, inversion, or manipulation of RAG-retrieved content.

Prevention

- Validate and sanitize data before embedding
- Use adversarial training or robustness checks on embeddings
- Apply access controls and versioning to vector stores

OWASP Top 10 LLM



Attackers extract or infer hidden system prompts, exposing proprietary instructions or jailbreak defenses.

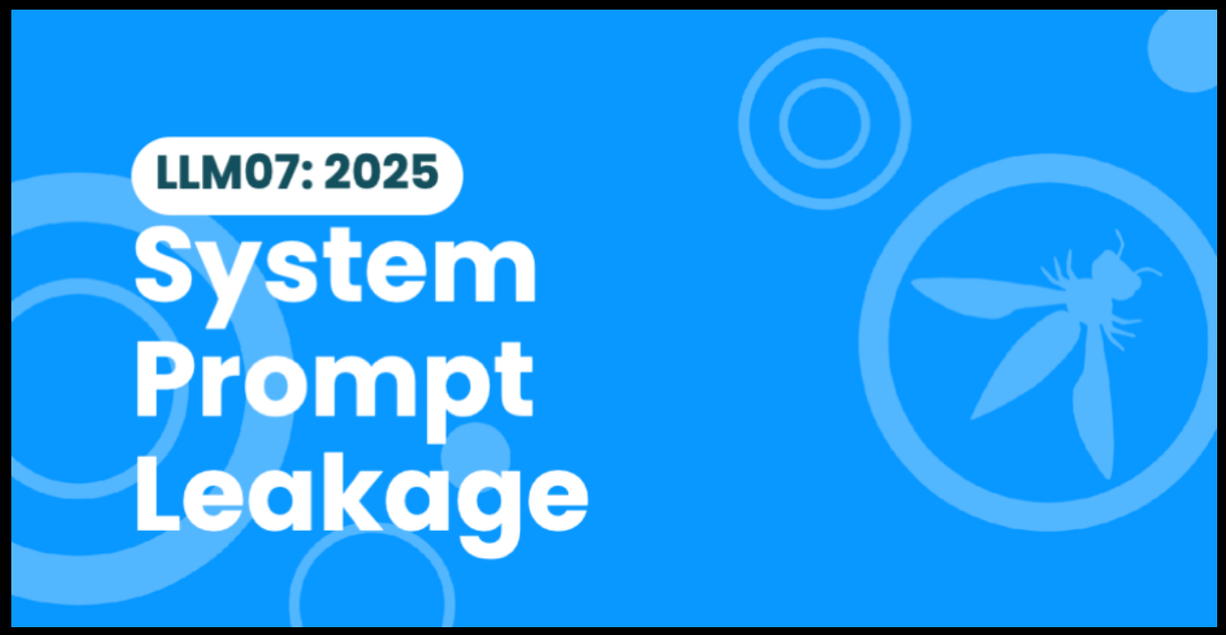
Threat

- Prompt leakage reveals internal guidelines or filters
- Extracted prompts enable easier jailbreaking
- Leaked prompts compromise competitive secrets

Example

https://www.artificial-intelligence.blog/ai-news/cybersecurity-and-llms
(references 2025 incidents of system prompt extraction in commercial LLMs)

OWASP Top 10 LLM



Attackers extract or infer hidden system prompts, exposing proprietary instructions or jailbreak defenses.

Prevention

- Obfuscate or encrypt system prompts where possible
- Use prompt guardrails and output filters to block extraction attempts
- Regularly rotate and version system instructions

OWASP Top 10 LLM

Granting LLMs too much autonomy to execute actions without oversight risks unintended or malicious consequences.

Threat

- LLM agent deletes files or sends unauthorized emails
- Autonomous tool calls escalate privileges unexpectedly
- Agent chains perform harmful multi-step actions

Example

https://www.invicti.com/blog/web-security/owasp-top-10-risks-llm-security-2025
(highlights emerging agentic AI incidents with excessive permissions)

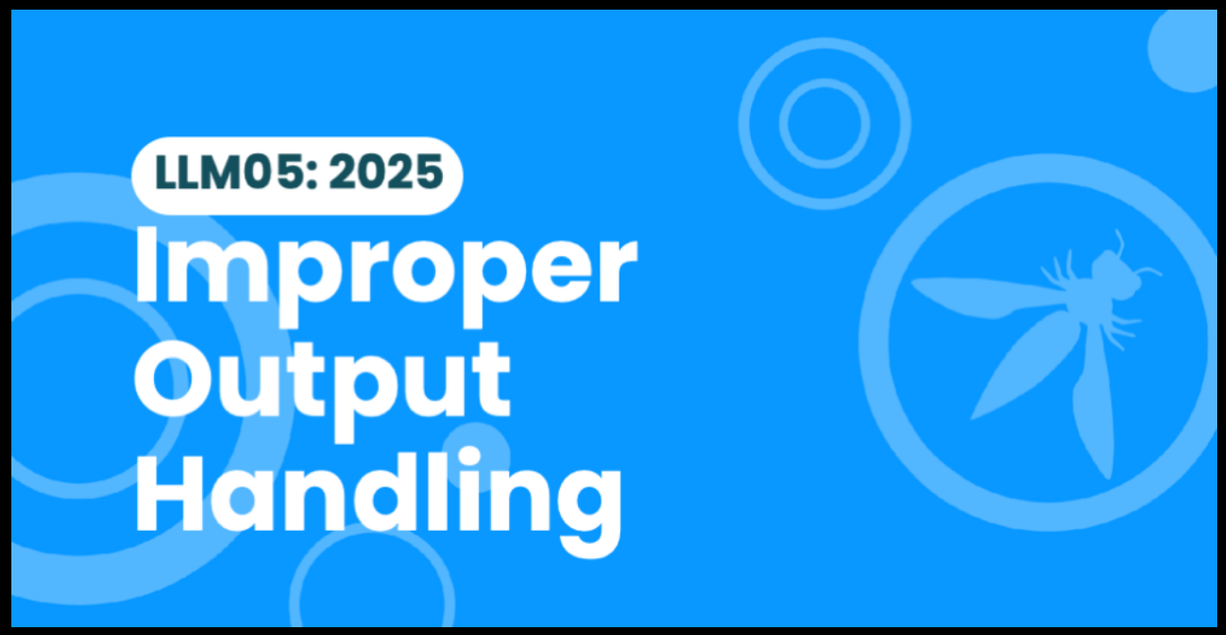
OWASP Top 10 LLM

Granting LLMs too much autonomy to execute actions without oversight risks unintended or malicious consequences.

Prevention

- Enforce least-privilege access for LLM-invoked tools/APIs
- Require user confirmation for sensitive or irreversible actions
- Sandbox agent executions with strict boundaries

OWASP Top 10 LLM



Failing to sanitize or validate LLM outputs before downstream use enables code execution, XSS, or other exploits.

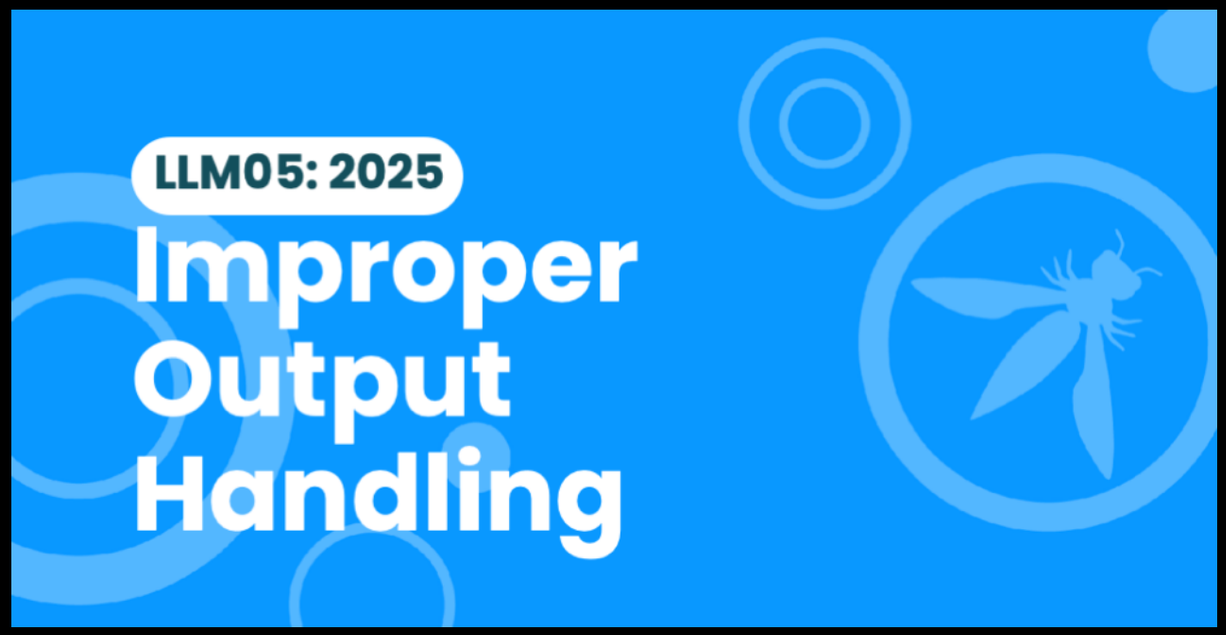
Threat

- Malicious output executes as code in backend
- Unsanitized response triggers XSS in web UI
- Output poisons integrated systems or logs

Example

https://data-security.blog/2024/05/11/safeguarding-ai-systems-against-prompt-injections
(includes output handling failures in prompt injection chains)

OWASP Top 10 LLM



Failing to sanitize or validate LLM outputs before downstream use enables code execution, XSS, or other exploits.

Prevention

- Sanitize and validate all LLM-generated content
- Use content security policies for rendered outputs
- Isolate LLM outputs in sandboxed environments

OWASP Top 10 LLM

LLMs unintentionally reveal confidential data from training or context, including PII or proprietary info.

Threat

- Model regurgitates personal data from training set
- Context leakage exposes user secrets in responses
- Inadvertent disclosure of internal docs via RAG

Example

https://www.oligo.security/academy/owasp-top-10-llm-updated-2025-examples-and-mitigation-strategies
(references data leakage incidents tied to model disclosure)

OWASP Top 10 LLM

LLMs unintentionally reveal confidential data from training or context, including PII or proprietary info.

Prevention

- Apply data sanitization and PII redaction filters
- Use differential privacy techniques in fine-tuning
- Enforce output scanning for sensitive patterns

OWASP Top 10 LLM

Compromised dependencies, datasets, models, or plugins undermine LLM integrity and introduce backdoors.

Threat

- Poisoned pre-trained model introduces hidden behaviors
- Vulnerable plugin executes arbitrary code
- Compromised dataset biases or backdoors model

Example

https://blog.qualys.com/vulnerabilities-threat-research/2024/11/25/ai-under-the-microscope-whats-changed-in-the-owasp-top-10-for-llms-2025
(covers supply chain compromises in LLM ecosystems)

L S E

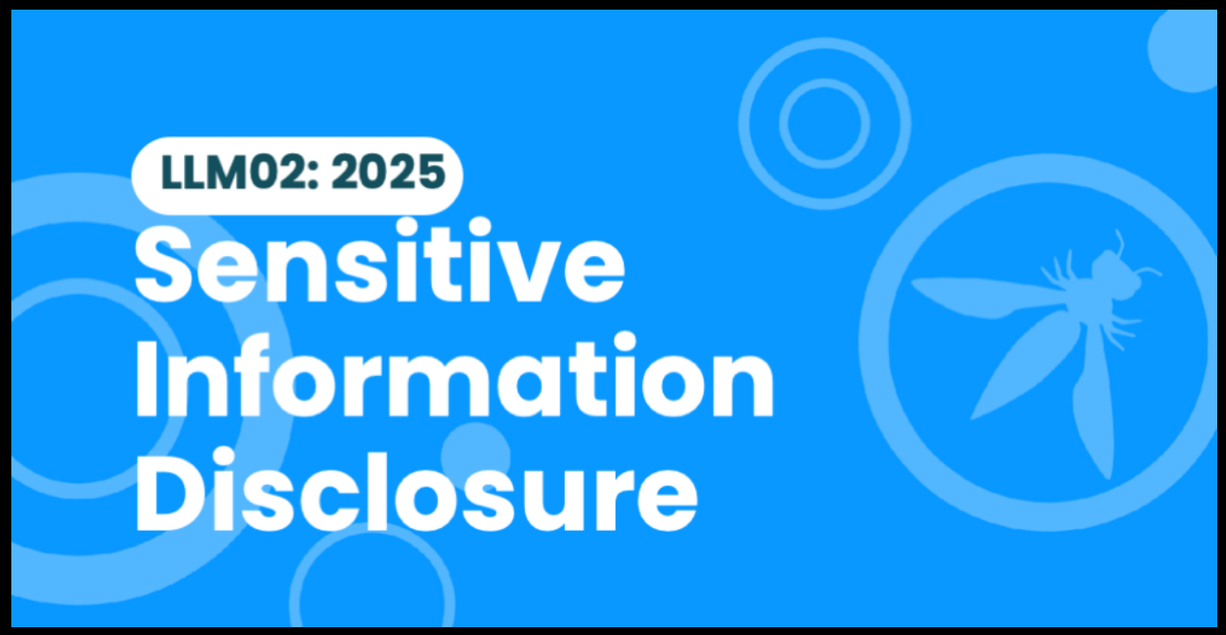
OWASP Top 10 LLM

Compromised dependencies, datasets, models, or plugins undermine LLM integrity and introduce backdoors.

Prevention

- Vet and verify all third-party models/datasets/plugins
- Use software bill of materials (SBOM) for tracking
- Apply integrity checks and signing on components

OWASP Top 10 LLM



Malicious tampering with training data or fine-tuning corrupts model behavior or injects backdoors.

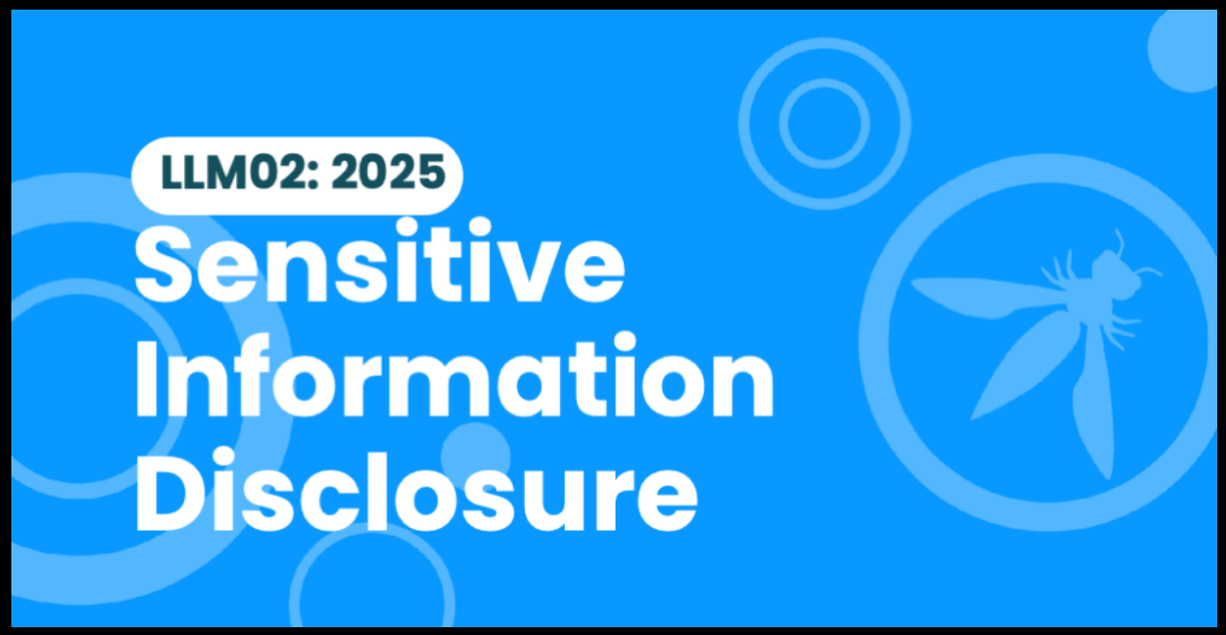
Threat

- Poisoned examples cause biased or harmful outputs
- Backdoor triggers activate on specific inputs
- Data tampering degrades model accuracy

Example

https://www.machine.news/llms-can-be-hypnotized-into-producing-poisoned-responses-ibm-and-mit-researchers-warn
(details real poisoning via feedback/upvotes in 2025 research)

OWASP Top 10 LLM



Malicious tampering with training data or fine-tuning corrupts model behavior or injects backdoors.

Prevention

- Validate and audit training/fine-tuning datasets
- Use anomaly detection on input data streams
- Apply robust training methods against poisoning

OWASP Top 10 LLM

Crafted inputs override LLM instructions, leading to unauthorized actions, data leaks, or policy violations.

Threat

- Jailbreak bypasses safety filters for harmful content
- Indirect injection via uploaded files or data
- Prompt overrides to exfiltrate sensitive data

Example

https://nickrogers.blog/locking-down-ai-essential-tips-for-securing-llm-applications
(demonstrates real prompt injection attacks bypassing defenses)
https://www.bankinfosecurity.com/deepseek-ai-models-vulnerable-to-jailbreaking-a-27428

OWASP Top 10 LLM

Crafted inputs override LLM instructions, leading to unauthorized actions, data leaks, or policy violations.

Prevention

- Enforce privilege separation between user and system prompts
- Use input/output guard models or filters
- Apply prompt delimiters and adversarial testing

Examples

DEMO

Summary

Where we can be hacked?

- Data Collection and Handling
  uploads, emails, RAG, etc

- Model development and training
  source data, web, emails, tickets, insider

- Model inference engine and
  model behavior

- Tools
  vulnerability within tools

- MCP components
  vulnerability within component

Observability Next Steps

Priority	Practice	Why It Matters	Implementation Tip
1	**Prompt Injection & Jailbreak Monitoring**	Detects 70%+ of LLM attacks (e.g., bypassing safety filters)	Use rule-based filters + fine-tuned models to flag: ;system;, --no-safety, escape patterns. Example tool: Langfuse's attack detection.
2	**Token Usage Monitoring**	Jailbreaks consume 5-10x more tokens than normal queries	Alert when token usage > 95% of avg. per session (e.g., 50k tokens = high-risk). Critical for cost control + threat detection.
3	**Semantic Consistency Metric**	Measures model drift (e.g., answers change when prompt is slightly modified)	Track variance in responses to semantically identical prompts across 10 sessions. Alert if variance > 15%.
4	**User Activity Clustering**	Identifies bot-like behavior (DDoS, data scraping)	Monitor: Session length > 10 mins, Active users spike > 20%, Repeats same prompt 5x. Use tools like phospho for clustering.
5	**Harmful Output Moderation**	Prevents legal / reputational damage (bias, hate speech, etc.)	Real-time filters for: offensive language, sensitive data exposure, misinformation. Must be in production before launch.

Threat Summary

- Excessive resource abuse leading to DoS, cost overruns, or model extraction

- Propagation of misinformation, hallucinations, or unverified outputs causing harm

- Over-autonomy in agents/tools resulting in unintended destructive actions

- Leakage or extraction of proprietary prompts, data, or model internals

- Poisoning, injection, or supply chain compromises corrupting model integrity

Prevention Summary

- Implement strict rate limits, quotas, monitoring, and resource caps

- Enforce input/output validation, sanitization, and guardrail filters

- Apply least-privilege, sandboxing, and human oversight for actions

- Use data auditing, integrity checks, and adversarial robustness techniques

- Ground responses in verified sources and cross-check critical outputs

Prevention Summary (cont)

- No to logging sensitive data
  use hashes for input and output data to detect duplicate requests, if you need to keep the data, put hashes in log and encrypt data, store outside the logs

- Use containers to host models internally
  lock them down, no root, no network outside internal system, require authentication and authorization, follow least privilege

- Smaller memory models have less re-cognition, less recall to training, so they are easier to bypass, they lack adversarial training

Testing Resources

SPIKEE
REVERSEC

DeepTeam.

LLM Red Teaming Framework

- ✓ Security Scanner
- ✓ LLM Guardrails
- ✓ LLM OWASP Top 10
- ✓ Multi + Single Turn

trulens

DeepEval.

ragas

JUnit

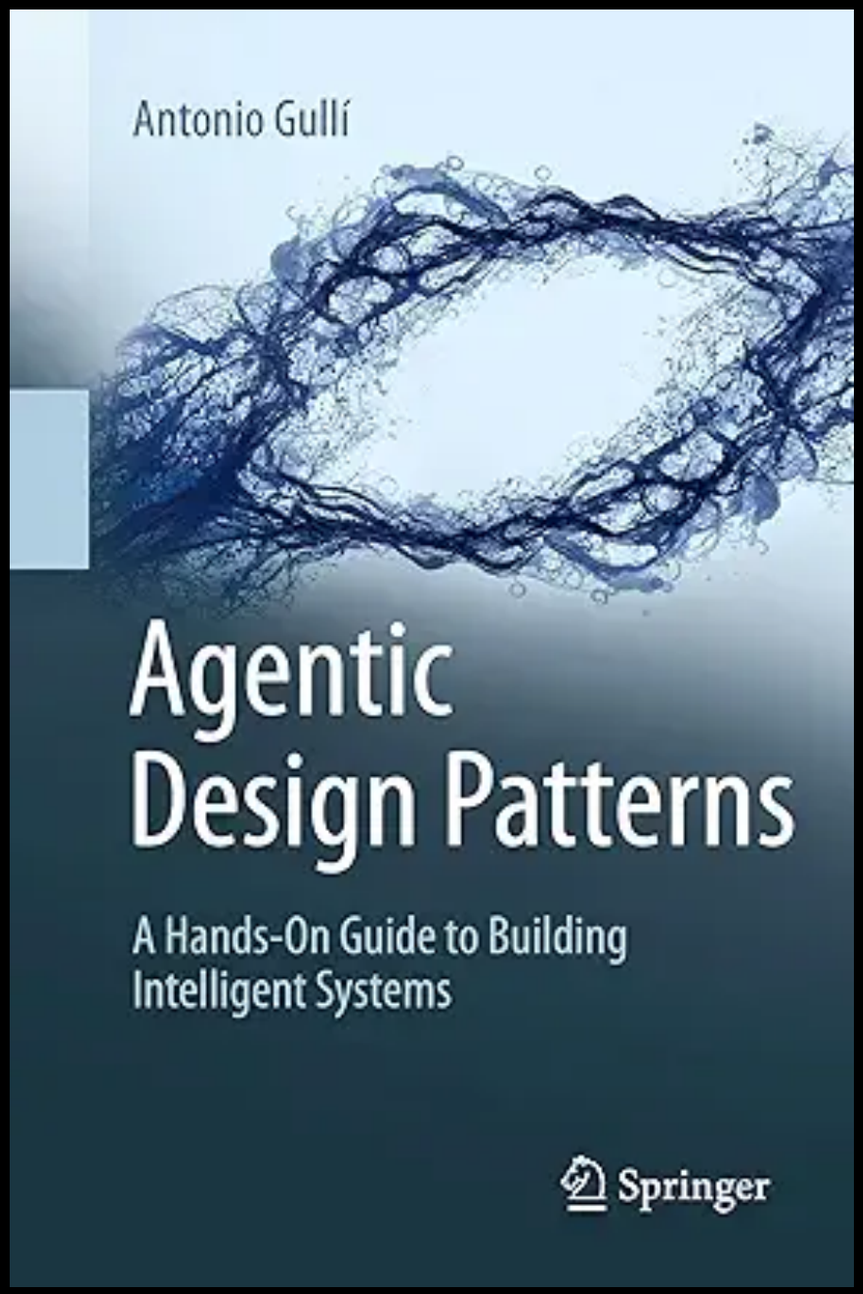
https://genai.owasp.org/initiatives/#ai-redteaming
https://github.com/requie/LLMSecurityGuide

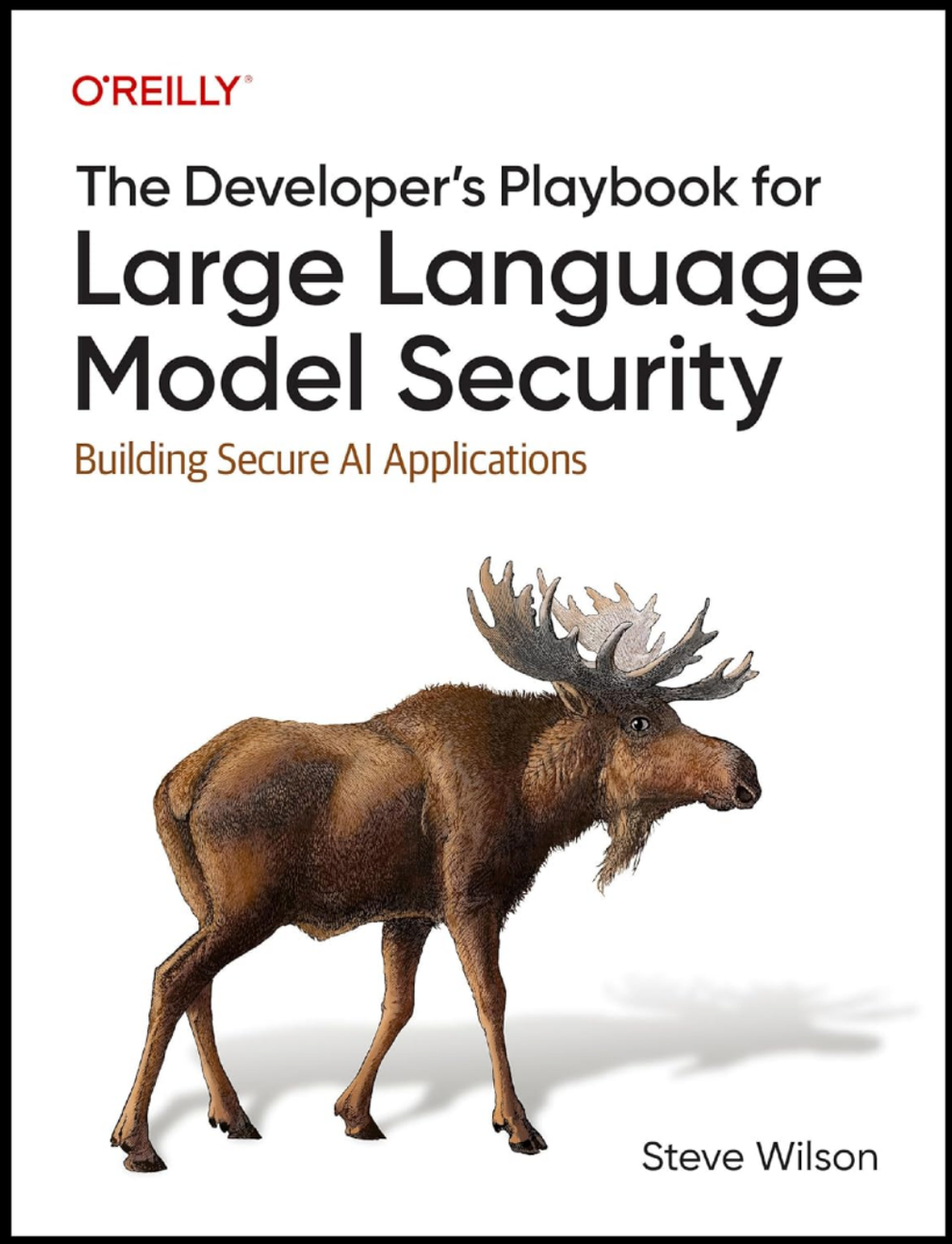
Resources

- https://genai.owasp.org/llm-top-10
- https://genai.owasp.org/resource/owasp-top-10-for-llm-applications-2025
- https://owasp.org/www-project-top-10-for-large-language-model-applications
- https://owasp.org/www-project-top-10-for-large-language-model-applications/assets/PDF/OWASP-Top-10-for-LLMs-v2025.pdf
- https://www.promptfoo.dev/docs/red-team/owasp-llm-top-10
- https://www.confident-ai.com/blog/owasp-top-10-2025-for-llm-applications-risks-and-mitigation-techniques
- https://blog.qualys.com/vulnerabilities-threat-research/2024/11/25/ai-under-the-microscope-whats-changed-in-the-owasp-top-10-for-llms-2025
- https://blog.barracuda.com/2024/11/20/owasp-top-10-risks-large-language-models-2025-updates

L S E

Gift Giveaway





LSE

Questions ?

https://github.com/lseinc/cm26-llm-lockdown.git

Don’t forget to fill out the survey!

LLM Lockdown: Defending Against OWASP's Top 10 AI Nightmares

LSE

Thank You !

https://github.com/lseinc/cm26-llm-lockdown.git

David Lucas
Lucas Software Engineering, Inc.
www.lse.com
ddlucas@lse.com
@DavidDLucas

LSE

https://github.com/lseinc/cm26-llm-lockdown.git

David Lucas
Lucas Software Engineering, Inc.
www.lse.com
ddlucas@lse.com
@DavidDLucas

L S E