OWASP LLM Top 10 Security Risks Report (2024)

Executive Summary

This report provides a comprehensive analysis of the ten most critical security risks associated with Large Language Model (LLM) applications, as identified by the OWASP LLM Top 10 Project for 2024. These risks represent a new attack surface that combines traditional web application vulnerabilities with AI-specific threats. Organizations deploying LLM-powered services must address these risks through a systematic approach encompassing secure development practices, robust deployment configurations, and continuous monitoring.

Introduction

Large Language Models have rapidly become integral to modern applications, from customer service chatbots to code generation assistants. However, this widespread adoption introduces unique security challenges that extend beyond conventional application security concerns. The OWASP LLM Top 10 framework provides a structured approach to understanding and mitigating these risks, enabling organizations to deploy LLM services securely while maintaining compliance with regulatory requirements.

Risk 1: Prompt Injection

Overview

Prompt injection occurs when untrusted user input becomes part of the system or assistant prompt, allowing attackers to alter the model's intended behavior. This vulnerability enables adversaries to manipulate the model into disclosing sensitive information, executing unauthorized commands, or producing prohibited content.

Impact Assessment

The consequences of successful prompt injection attacks include confidentiality breaches, integrity violations, credential leakage, and execution of malicious actions through tool-calling mechanisms. In enterprise environments, this can lead to unauthorized access to internal systems and data exfiltration.

Real-World Attack Scenarios

Organizations have experienced prompt injection attacks across various use cases. In customer-facing chatbots, attackers have used phrases like "Ignore previous directions" to extract administrative credentials embedded in context. Code generation assistants have been manipulated to produce unsafe code by injecting comments that remove safety checks. RAG-enabled search systems have been exploited to retrieve sensitive file contents through carefully crafted queries.

Mitigation Strategy

Effective defense against prompt injection requires multiple layers of protection. Organizations should maintain strict separation between control instructions and user input, never concatenating raw user text into system prompts. Input validation through whitelisting and schema validation provides an essential first line of defense. Implementing instruction-only prompts keeps system prompts static while placing user text exclusively in user roles. Post-processing filters using safety classifiers should examine model outputs before returning them to users. When tool-calling functionality is enabled, organizations must validate that generated commands conform to predefined allow-lists. Regular red-team testing using known jailbreak corpora ensures ongoing protection against evolving attack techniques.

Risk 2: Data Poisoning (Training / Fine-tuning)

Overview

Data poisoning attacks involve the manipulation of training or fine-tuning datasets to inject malicious patterns or backdoors into the model. These attacks can cause the model to exhibit specific unintended behaviors when triggered by particular input patterns.

Impact Assessment

Poisoned models may produce biased outputs, execute backdoored behaviors, or systematically misclassify inputs in ways that benefit attackers. The effects can persist throughout the model's deployment lifecycle and affect all downstream users.

Real-World Attack Scenarios

Documented examples include backdoor triggers where thousands of malicious training examples associate benign terms with incorrect classifications, causing the model to systematically misclassify legitimate content. In code generation contexts, attackers have embedded destructive commands within code snippet datasets, resulting in models that suggest harmful operations.

Mitigation Strategy

Organizations must establish robust data provenance and integrity controls, including cryptographic hashing of all ingested files and provenance metadata storage. Automated dataset sanitization should remove duplicates, detect unusual token distributions, and filter toxic language beyond policy thresholds. Human review processes should validate data from external partners. Differential privacy training techniques add noise to gradients to limit memorization of rare examples. Backdoor detection procedures test sample triggers to ensure outputs remain benign.

Comprehensive versioned model cards track the exact relationship between dataset versions and resulting models.

Risk 3: Model Extraction / Intellectual Property Theft

Overview

Model extraction attacks involve adversaries making repeated queries to reconstruct a close approximation of proprietary model weights or behavior patterns, effectively stealing intellectual property through systematic probing.

Impact Assessment

Successful extraction attacks result in competitive losses, intellectual property exposure, and potential downstream abuse of reconstructed models. Organizations lose the competitive advantage derived from their model investments.

Real-World Attack Scenarios

Research has demonstrated that attackers can reconstruct large parameter models with high similarity using systematic querying strategies. API crawling bots that vary parameters and record responses enable training of clone models that replicate proprietary behavior.

Mitigation Strategy

Defense requires multiple concurrent controls. Rate limiting and quota systems restrict per-API-key requests and enforce exponential backoff on repeated identical queries. Response randomization introduces controlled variance so identical prompts rarely produce identical outputs. Watermarking embeds invisible signatures in outputs that can prove ownership. Legal protections through terms and conditions explicitly prohibit model recreation, supported by automated abuse reporting systems. Query behavior monitoring flags high-entropy patterns characteristic of extraction attempts.

Risk 4: Sensitive Data Leakage (Prompt/Response)

Overview

LLMs can memorize and reproduce snippets of training data or echo private information supplied in prompts, resulting in privacy regulation violations and unauthorized disclosure of personally identifiable information, secrets, or proprietary data.

Impact Assessment

Data leakage incidents can trigger GDPR and CCPA violations, credential compromise, and regulatory penalties. Organizations may face legal liability and reputational damage from unauthorized disclosure.

Real-World Attack Scenarios

Documented incidents include cases where models reproduced exact code from private repositories, including embedded secrets, when prompted with partial snippets. Corporate chatbots have completed partially redacted identifiers using memorized training data, exposing customer information.

Mitigation Strategy

Organizations should implement comprehensive data redaction for both inputs and outputs using regular expressions, named entity recognizers, and specialized data loss prevention tools. Fine-tuning processes should incorporate forget capabilities using gradient-based unlearning techniques. When strong guarantees are required, differential privacy trained models provide mathematical privacy protections. Output filtering through policy engines screens responses before user delivery. User-level access controls restrict unstructured prompts based on clearance levels and need-to-know principles.

Risk 5: Unauthorized Access & Privilege Escalation

Overview

Inadequate authentication and authorization controls allow unauthenticated users to access privileged LLM endpoints or enable token reuse across contexts, permitting unauthorized execution of privileged functions.

Impact Assessment

Exploitation leads to data exfiltration, system takeover, unauthorized provisioning of resources, and compromise of downstream systems accessible through LLM tool-calling mechanisms.

Real-World Attack Scenarios

Organizations have inadvertently exposed completion endpoints without API key requirements, enabling unlimited content generation. Misconfigured role-based access has granted junior developer tokens the ability to invoke administrative tools with broad system privileges.

Mitigation Strategy

Zero-trust API gateway architectures enforce OAuth2 or JWT validation on every request. Scope-based tokens separate permissions for completion, embedding, and administrative actions. IP allow-listing and VPC isolation keep internal APIs behind private networks with strictly controlled external facades. Multi-factor authentication protects privileged operations including prompt editing, fine-tuning, and external tool invocation. Regular secret rotation and immediate revocation procedures respond to compromise indicators.

Risk 6: Hallucination / Misinformation

Overview

Models generate plausible but factually incorrect statements that can cause severe harm in contexts requiring accuracy such as legal advice, medical triage, financial guidance, and code generation.

Impact Assessment

Hallucinations result in reputational damage, legal liability from incorrect advice, and failures in automated systems that depend on model outputs. In regulated industries, misinformation can trigger compliance violations.

Real-World Attack Scenarios

Legal assistants have generated fictitious court rulings that led to frivolous lawsuits. Code completion systems have produced syntactically correct but semantically flawed functions that compiled successfully but caused production failures.

Mitigation Strategy

Retrieval Augmented Generation grounds every answer in verified knowledge sources with explicit source citations. Confidence scoring returns calibrated probabilities alongside textual answers, enabling fallback mechanisms for low-confidence responses. Post-processing verification uses factuality checkers or external APIs to validate claims before presentation. User education through prominent disclaimers and source-showing interfaces manages expectations. Continuous evaluation maintains benchmark metrics like TruthfulQA scores, triggering retraining or prompt adjustments when regressions occur.

Risk 7: Adversarial Prompting (Prompt Jailbreaks)

Overview

Sophisticated phrasing techniques cause models to ignore safety guardrails and produce disallowed content despite system-level prohibitions. These attacks evolve continuously as new bypass methods emerge.

Impact Assessment

Successful jailbreaks enable policy evasion and generation of illicit content, potentially exposing organizations to legal liability and reputational harm from association with prohibited outputs.

Real-World Attack Scenarios

The "Do Anything Now" jailbreak and similar techniques have demonstrated that carefully crafted prompts can override safety constraints. Chain-of-thought attacks manipulate the model's reasoning process to arrive at prohibited conclusions through incremental steps.

Mitigation Strategy

Multi-layer safety architectures apply both pre-prompt guardrails and post-response classifiers. Dedicated jailbreak detection models, fine-tuned on known attack corpora, intercept malicious prompts before they reach primary LLMs. Prompt sanitization strips or replaces command phrases like "ignore instructions" and "pretend you are." Dynamic guardrails rotate or randomize system prompts per session to prevent static bypass techniques. Regular red-team exercises using the latest jailbreak repositories ensure detection coverage exceeds 95 percent.

Risk 8: Insecure Deployment Configuration

Overview

Misconfigured hosting environments including containers, serverless functions, and managed services expose secrets, enable code injection, or operate with excessive privileges that amplify the impact of other vulnerabilities.

Impact Assessment

Configuration weaknesses lead to service disruption, remote code execution, data theft, and container escape scenarios that compromise host systems.

Real-World Attack Scenarios

Docker images with hardcoded API keys have granted unauthorized access to commercial services. Kubernetes pods running with privileged flags have enabled container escape and host filesystem access. Missing TLS implementations have exposed plaintext traffic to man-in-the-middle attacks.

Mitigation Strategy

Secrets management systems like Vault or KMS store sensitive credentials with runtime injection through environment variables rather than baked-in values. Leastprivilege container configurations run as non-root users with dropped Linux capabilities and read-only root filesystems. Network hardening enforces mutual TLS between components and denies unnecessary outbound internet access. CI pipeline integration of image scanning tools like Trivy identifies vulnerable packages and exposed secrets. Infrastructure as Code linting with Checkov or tfsec catches misconfigurations including open security groups and missing encryption. Automated patch management applies OS and kernel updates through rolling update windows with weekly container rebuilds.

Risk 9: Insufficient Logging, Monitoring & Auditing

Overview

Inadequate observability prevents detection of prompt injection attempts, data exfiltration, model abuse, and other security incidents. Without proper logging, forensic analysis and incident response become impossible.

Impact Assessment

Logging gaps enable persistent compromise and delay incident response, increasing the severity and duration of security breaches. Regulatory non-compliance may result from inability to produce required audit trails.

Real-World Attack Scenarios

Organizations without request identifiers cannot correlate suspicious outputs with originating requests during incident investigation. Missing audit trails for fine-tuning operations prevent answering regulatory inquiries about model provenance and data usage.

Mitigation Strategy

Structured request logging captures essential metadata including hashed user identifiers, API keys, IP addresses, prompt and response hashes, timestamps, latency measurements, and model versions. Security event alerts trigger on patterns including authentication failures, jailbreak indicators, and PII detection. Central log storage forwards events to tamper-evident SIEM systems using OpenTelemetry or similar protocols. Retention policies maintain logs for minimum 90 days with write-once read-many protections. Quarterly internal audits verify log completeness and integrity, with external auditor validation for compliance-intensive domains.

Risk 10: Governance & Compliance Gaps

Overview

Absent or inadequate governance frameworks result in unchecked bias, unlicensed data usage, and failure to adhere to sector-specific regulations including HIPAA, GDPR, FINRA, and other applicable standards.

Impact Assessment

Governance failures lead to regulatory fines, brand damage, legal exposure from copyright infringement or discrimination claims, and systematic deployment of biased or harmful models.

Real-World Attack Scenarios

Medical chatbots trained on copyrighted journal articles have faced publisher lawsuits. Hiring AI systems lacking training data documentation have triggered regulatory investigations for potential disparate impact violations.

Mitigation Strategy

Comprehensive model cards document provenance, intended use, performance metrics, bias evaluation results, and known limitations. Legal review processes verify licensing for all datasets with maintained registries of sources, licenses, and expiration dates. Bias and fairness testing using standardized frameworks like Aequitas or Fairlearn examines protected attributes before deployment. Risk assessment lifecycles conduct threat modeling at design, pre-release, and post-update stages. Incident response playbooks predefine procedures for data leaks, jailbreaks, and model extraction including notification obligations. Regulatory mapping aligns each model's data flow with applicable regulations and implements required controls.

Implementation Framework

Self-Assessment Methodology

Organizations should conduct systematic security assessments using a structured approach. Begin by inventorying every LLM-powered component including APIs, user interfaces, internal tools, and autonomous agents. For each component, evaluate all ten risk categories with severity ratings. Prioritize remediation using a risk matrix combining likelihood and impact, addressing highest-scoring items first. Document all implemented controls with deployment dates. Execute automated validation including unit tests for input sanitization, prompt fuzzing using LLM-Fuzz toolkit, and load testing with rate limit enforcement. Enable SIEM dashboards monitoring the five most common alert types: jailbreak attempts, PII leakage, extraction spikes, authentication failures, and configuration drift. Establish quarterly governance reviews examining new LLM releases, dataset additions, and regulatory changes.

Quick-Win Actions

Organizations can achieve immediate security improvements through targeted interventions. For prompt injection defense, wrap user text in static assistant roles with whitelist filters. Address data poisoning by hashing and signing all training files with rejection of unverified data. Prevent model extraction through per-key request quotas and random temperature jitter. Mitigate data leakage using PII redaction engines on all outputs. Strengthen access control with OAuth2 enforcement and per-endpoint scopes with weekly key rotation. Reduce hallucination risk by grounding answers in RAG stores with source citations. Deploy jailbreak detector models in front of primary LLMs. Secure deployments by scanning Docker images for secrets and enabling mutual TLS. Improve observability by shipping structured request logs to SIEM systems with 180-day retention. Address governance gaps by publishing model cards and conducting bias audits before releases.

Conclusion

The OWASP LLM Top 10 provides a comprehensive framework for securing LLM-powered applications against both traditional web vulnerabilities and AI-specific threats. Organizations must treat LLM services as new attack surfaces requiring dedicated security controls integrated throughout development pipelines, CI/CD processes, and operational monitoring. By systematically addressing each risk category through the recommended mitigation strategies, organizations can deploy LLM capabilities while maintaining security, privacy, and regulatory compliance. The rapidly evolving nature of LLM technology demands continuous vigilance, regular security assessments, and adaptation of controls to emerging threat patterns.

References and Resources

- OWASP LLM Top 10 Project (2024): https://owasp.org/www-project-llm-top-10/
- OpenAI Security Best Practices: https://platform.openai.com/docs/guides/security
- Prompt Injection Detection Dataset: https://github.com/Azure/Prompt-Inject-Dataset
- RAG Boilerplate (LangChain + Pinecone): https://github.com/hwchase17/langchain
- LLM-Fuzz Toolkit: https://github.com/llm-fuzz/llm-fuzz
- Differential Privacy for LLMs (Google DP-SGD): https://github.com/google/differential-privacy

Appendix: Risk Summary Table

#	Risk	Impact	Priority Mitigation
T1	Prompt Injection	Confidentiality/integrity breach, credential leakage	Instruction-only prompts, sandboxing, input sanitization
T2	Data Poisoning	Model misbehavior, backdoors, bias	Data provenance, vetting, anomaly detection
T3	Model Extraction	IP exposure, competitive loss	Rate limiting, response randomization, watermarking
T4	Sensitive Data Leakage	GDPR/CCPA violations, credential compromise	Output filtering, differential privacy, redaction
T5	Unauthorized Access	Data exfiltration, system takeover	Zero-trust API keys, RBAC, MFA
T6	Hallucination	Reputation damage, legal liability	RAG, verification, confidence scores
T7	Adversarial Prompting	Policy evasion, illicit content	Multi-layer safety, jailbreak detection
T8	Insecure Deployment	Service disruption, RCE, data theft	IaC hardening, secret management, TLS
T9	Insufficient Logging	Persistent compromise, delayed response	Structured logs, audit trails, alerts
T10	Governance Gaps	Regulatory fines, brand damage	Model cards, bias testing, legal review