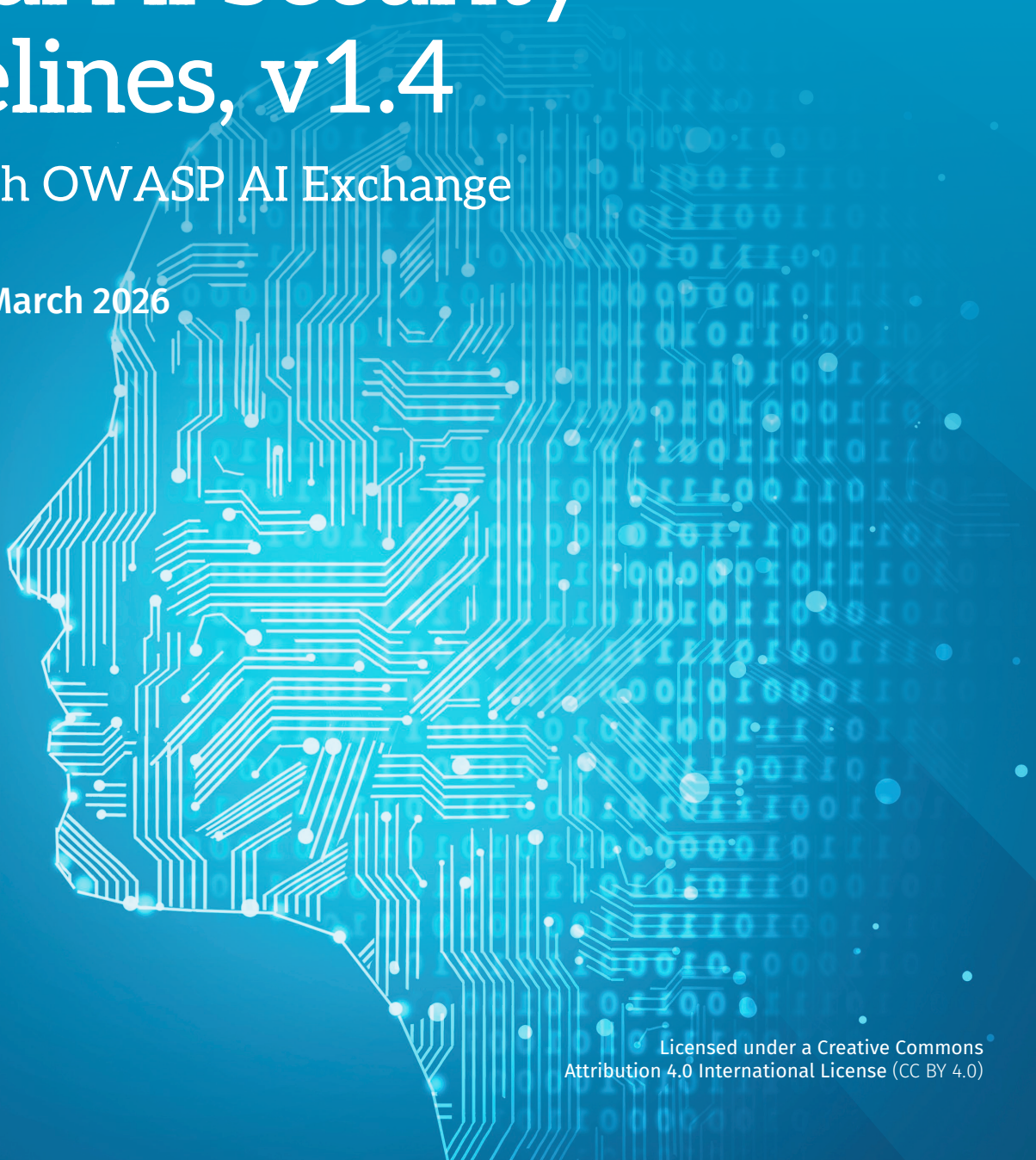
SANS | Research Program

Critical AI Security Guidelines, v1.4

Aligned with OWASP AI Exchange

Last updated: March 2026



Contributing Authors

- **Ahmed Abugharbia**, Fortinet
- **Sarthak Agrawal**, US Congress
- **Brett Arion**, Binary Defense
- **Matt Bromiley**, Prophet Security
- **Ben Buchanan**, Biden White House AI Advisor
- **Chris Cochran**, SANS Institute
- **Ron F. Del Rosario**, SAP
- **Mick Douglas**, InfoSec Innovations
- **David Hoelzer**, Occulumen
- **Ken Huang**, DistributedApps.ai
- **Bhavin P. Kapadia**, Stablecoins
- **Rob Lee**, SANS Institute
- **Seth Misenar**, Context Security Consulting
- **Helen Oakley**, SAP
- **Jorge Orchilles**, Verizon
- **Jason Ross**, OWASP AI
- **Rakshith Shetty**, Palo Alto Networks
- **Jim Simpson**, HiddenLayer
- **Jochen Staengler**, BSI
- **Rob van der Veer**, Software Improvement Group
- **Jason Vest**, Binary Defense
- **Eoin Wickens**, Hidden Layer
- **Manish Kumar Yadav**, SAP
- **Sounil Yu**, Knostic AI

Overview

As artificial intelligence (AI) continues to revolutionize enterprises and security practices, ensuring security-focused controls for AI implementations is paramount. This document has been developed in alignment with the OWASP AI Exchange to provide uniform, comprehensive, actionable security guidance.

The control categories in this document are:


Conventional Security Controls (e.g., Access Controls and Data Protection)


Data Minimization and Obfuscation


AI Supply Chain Management


Data/Model Engineering Controls


Limit Model Behavior


Model I/O Handling


Monitoring


Governance, Risk, Compliance (GRC)

The OWASP AI Exchange is a formal partner of SANS Institute and provides complementary guidance on threats and controls at owaspai.org.

Control Categories

As organizations incorporate AI into their operations, they must seek and adopt comprehensive security strategies to mitigate risks. The following sections explore key AI control categories, providing detailed recommendations for secure implementation.

Conventional Security Controls

This combined category covers all traditional security controls applied to AI assets, enabling organizations to include AI systems within their existing Information Security Management System (ISMS) in a straightforward fashion.

Control access to interaction/inference

Depending on the use case and deployment of the AI system, authentication and access controls should be implemented where appropriate. For external-facing applications, the front-end must authenticate to the AI system back-end.

Internal AI systems, or those containing sensitive data, should be used with authentication and access controls. Unless absolutely necessary, unauthenticated and/or unmonitored access to AI system APIs or front-ends should not be allowed.

API keys should be properly managed under robust, secure software development policies.

See Monitoring on how to monitor interaction/inference.

References

- OWASP AI Exchange on Model access control

Protect Your Model Parameters

It is critical to ensure traditional security controls, such as principle of least privilege and strong access controls with accountability, have been implemented to protect models. Should an unauthorized individual be able to replace or modify a deployed model, untold damage can result.

Consider a generative agentic system leveraging a large language model (LLM). If an attacker were to tamper with the model or prompt for an auditor agent that acts as a gatekeeper, it may suddenly become possible to cause inappropriate responses to be generated because the auditor has been subverted.

In addition to traditional access controls, protecting model parameters requires applying additional layers of defense. Techniques such as encryption of model files at rest, runtime obfuscation, and the use of trusted execution environments (TEEs) can reduce the risk of unauthorized access or model exfiltration. For further guidance, refer to OWASP AI's resource on runtime model theft.

If attackers gain unauthorized access to model parameters, it can enable them to prepare input attacks by analyzing the parameters and/or experimenting on the model without being restrained or noticed.

References:

- OWASP AI Exchange on development-time model leak
- OWASP AI Exchange on development-time model poisoning
- OWASP AI Exchange on runtime model leak
- OWASP AI Exchange on runtime model poisoning

Protecting Augmentation Data

Especially in generative AI, input to models can be augmented with additional data, from the context, system prompts, or retrieved by a retrieval-augmented generation (RAG) architectures. In such architectures, vector databases are commonly used to store and retrieve semantically indexed data that is fed into LLMs. However, augmentation data used in these systems can be a source of significant risk if not properly secured.

Augmentation data should be treated as sensitive, especially if it influences LLM responses. If tampered with, this data can cause models to generate misleading or dangerous outputs. In addition to enforcing least privilege access models for both read and write operations, organizations should implement secure upload pipelines, logging and auditing of changes, and validation mechanisms to detect unauthorized modifications.

References

- OWASP AI Exchange on leaking sensitive augmentation data
- OWASP AI Exchange on manipulating augmentation data

Defend Training Data

Models are only as good as their training data. Adversarial access can negatively impact training data, hiding malicious activities. This applies not just to LLMs, but to any model that will make security or operational decisions.

Referencess

- OWASP AI Exchange on data poisonings
- OWASP AI Exchange on train and test data leaking

Avoid Data Commingling

Leveraging enterprise data allows for better grounded applications. Sensitive data should be sanitized or anonymized prior to LLM incorporation.

Data Minimization and Obfuscation

This control category highlights a critical AI-specific consideration: Because of zero model trust, the prevention of data leakage is particularly important for AI systems. Additionally, because AI systems work with stochastic data, new opportunities for obfuscation exist that are not available in conventional systems.

Minimize Data Exposure

Apply data minimization principles throughout the AI lifecycle. Only provide models with the minimum data necessary for their intended function. This reduces the attack surface and limits potential damage from data leakage.

Limit Sensitive Prompt Content

Attackers with unauthorized access to an organization's prompts can infer sensitive information such as internal business processes, proprietary data, decision logic, or personally identifiable information (PII). Avoid including sensitive or confidential information in prompts whenever possible.

Leverage AI-Specific Obfuscation Techniques

AI systems can leverage unique obfuscation methods:

- **Differential privacy—**Add calibrated noise to training data or model outputs to prevent inference of individual data points
- **Data distortion—**Apply transformations to input data that preserve utility for the model while obscuring sensitive details
- **Synthetic data generation—**Train models on synthetic data that preserves statistical properties without exposing real data
- **Federated learning approaches—**Train models across distributed datasets without centralizing sensitive data

Context Window Management

Be mindful of what data enters the context window during inference. Implement controls to prevent sensitive data from being inadvertently included in prompts or retrieved context.

References

- OWASP AI Exchange on data minimization and obfuscation
- OWASP AI Exchange on federated learning

AI Supply Chain Management

Supply chain management of data and models, and dealing with suppliers that host your model, deserves special attention as a control category. This is a particularly important consideration for AI systems given the prevalence of public models, shared datasets, and third-party hosting arrangements.

Assess Model Hosting Options: Local vs. SaaS Models

There are several models available that can be hosted locally, which is beneficial for use cases where data privacy is critical and sharing with a third party is not desirable. Hosting these LLMs locally ensures greater control over the data, but the trade-off is the need for sufficient processing power to run and manage them effectively. Furthermore, locally hosted models may not have good reasoning performance compared with frontier models.

When weighing where and how to host AI solutions, be sure to think carefully about and codify legal requirements in any contracts. For example: Will your data ever be used or retained by the provider for training or refining a model? If the provider claims that they will not store or use your data, what steps have been taken to prevent your data from being logged?

Be Cautious Using Public Models

Sites such as HuggingFace are wonderful resources for datasets, models, and various tools to facilitate rapid development of AI-based solutions. However, caution is required when using them. Some of the mechanisms used to share models can be leveraged by bad actors to introduce malicious code into the packaging used to deploy the model.

Models also may be created by bad actors with architectural backdoors in them. Once a backdoor is created inside of a model, it can be difficult, if not impossible, to remove it via fine-tuning. Manually red team all imported models. Host vetted models in an internal model garden for developers to easily obtain.

Critical Models and Transfer Attacks

While sharing knowledge and models is laudable, sharing a trained model can introduce significant risk, especially for a model relied upon for security or operational decisions. A bad actor given a copy of your trained model can experiment with it to understand how to cause the model to misbehave.

Important nuance: If your model operates in a domain where equivalent public models exist, those models can be used for transfer attacks regardless of whether you share yours. A critical model may itself be a public one. The security posture should account for this reality.

AI Bill of Materials (AIBOM)

LLM applications depend upon a complex underlying ecosystem for their functionality. Modeled after software bill of materials (SBOM), creation and maintenance of an AIBOM can provide better visibility into relevant aspects of the AI supply chain, including considerations of dataset and model provenance. AIBOMs contain technical details that are useful to adversaries in attacking AI applications. Care should be taken to limit the disclosure of AIBOMs.

References

- OWASP AI Exchange on supply chain management for AI
- OWASP AI Exchange on supply-chain model poisoning
- OWASP AI Exchange on using ready-made models

Data/Model Engineering Controls

This category covers security techniques that AI engineers apply during model development. These controls are distinct from operational security controls because they are applied by AI/ML practitioners during the model lifecycle.

Adversarial Training and Robustness

Train models to be robust against adversarial inputs by including adversarial examples in training data. This helps models maintain correct behavior when faced with deliberately crafted malicious inputs.

- **Evasion-robust models—**Train models to resist adversarial perturbations designed to cause misclassification
- **Poison-robust models—**Implement defenses against training data poisoning attacks
- **Adversarial robust distillation—**Transfer robustness properties from larger to smaller models

References:

- OWASP AI Exchange on adversarial training
- OWASP AI Exchange on evasion robust models
- OWASP AI Exchange on distillation

Model Alignment and Fine-Tuning

Ensure models are aligned with intended behavior through:

- **Model alignment techniques—**RLHF, Constitutional AI, and other alignment methods
- **Fine pruning—**Remove unnecessary model capabilities that could be exploited
- **Continuous validation—**Regularly test model behavior against expected outputs

References

- OWASP AI Exchange on model alignment
- OWASP AI Exchange on poisoning-robust models
- OWASP AI Exchange on continuous validation

Data Quality Controls

- **Data quality control—**Validate and clean training data to prevent quality degradation
- **Training data distortion—**Apply controlled modifications to training data to improve robustness

References

- OWASP AI Exchange on data quality control

Architectural Considerations

- **Model ensemble—**Use multiple models to improve robustness and detect anomalies
- **Smaller models where appropriate—**Reduced complexity can mean reduced attack surface

References

- OWASP AI Exchange on model ensemble
- OWASP AI Exchange on smaller models

Engineering Considerations

The following factors should be evaluated during AI system design:

- **Modality—**Multimodal implementations can increase the attack surface. Safety and alignment can prove inconsistent across different modalities.
- **Languages and character sets—**Multilingual models can introduce vulnerabilities. Alignment mechanisms are often tailored to the most prominent training language.
- **Encoding/decoding—**Models often handle Base64, Hex, or Morse-encoded data without explicit instruction. Encoded prompt/response data might bypass security measures.
- **Compression/decompression—**Another means of input/output obfuscation available to adversaries. Support varies across model implementations.

Limit Model Behavior

In light of zero model trust, limiting what a model can do is an essential aspect of AI security. This control category addresses guardrails, access controls outside the model, focused functionality, and human oversight.

Establish AI Guardrails

Guardrails are rules that instruct a model on how to respond or avoid responding to specific topics. They can be created manually by searching for explicit values in the prompt or response, built in by the AI-hosting provider, or integrated with other AI that detect "trickery" attempts.

Even with guardrails, recognize that you cannot rely upon them to be infallible. Bad actors are notoriously good at coming up with creative ways to convince an AI model to do things its guardrails specifically prohibit. If there is information or actions your AI should never disclose or take, ensure your model does not have access to that information.

Implement Access Controls Outside of the Model

Many organizations are trying to leverage LLMs for knowledge retrieval where different users have different access rights. Attempting to implement these types of controls within the model is error prone and can often be easily subverted.

Instead, consider a RAG-style approach with access control lists (ACLs) applied in the vector retrieval system from which responses are generated. This eliminates the need to attempt to implement these guardrails in the LLM and has the benefit of limiting hallucinations.

Pay close attention to the use of function calling, especially in agentic AI systems. If not properly scoped, function calls may allow models to invoke external tools or actions beyond their intended purpose.

Employ the Principle of Focused Functionality

Despite continuous model evolution, AI applications should offer as limited functionality as is acceptable. "The worst enemy of security is complexity." In designing agents, explicitly define and limit the functions and tools the agent requires access to. Avoid assigning multiple tools to an agent and apply the principle of least privilege.

Human Oversight

Implement human oversight mechanisms appropriate to the risk level of AI decisions:

- **Human-in-the-loop—**Require human approval for high-stakes decisions
- **Human-on-the-loop—**Enable human monitoring and intervention capability for moderate-risk operations
- **Escalation mechanisms—**Define clear criteria and processes for escalating uncertain or high-risk AI outputs to human reviewers
- **Override capabilities—**Maintain the ability to override or halt AI operations when necessary

References

- OWASP AI Exchange on limiting unwanted behaviour

Model I/O Handling

Model I/O handling focuses on filtering and detecting input and output to protect against adversarial manipulation and unauthorized interactions during runtime operation.

Sanitize, Validate, and Filter Inputs/Prompts

Prompt injection represents the most common LLM application attack vector and warrants a multilayered approach to protection and detection. All prompts should be preprocessed prior to inference and all model outputs postprocessed prior to response. If employing RAG, additional LLM input filtering and validation would need to occur after the prompt has been augmented.

In multiuser environments or applications where prompts are composed from multiple sources, input segregation is critical. By tagging or isolating user-provided inputs from system-generated context, organizations can reduce the risk of indirect prompt injection.

Outside of generative AI, model input also requires detection of potential attack patterns, ranging from anomaly detection to observing suspicious series of inputs.

References

- OWASP AI Exchange on prompt injection I/O handling
- OWASP AI Exchange on anomalous input handling
- OWASP AI Exchange on evasion input handling
- OWASP AI Exchange on unwanted input series handling
- OWASP AI Exchange on segregating untrusted input data

Sanitize, Validate, and Filter AI Outputs/Responses

Although trying to prevent or detect prompt injection attempts should be considered necessary, the complexity and nuance of LLM applications make it obvious that merely controlling the input should not be considered sufficient. Much as validation and filtering of inputs proves vital, so too is properly handling and assessing outputs.

Keep in mind that multiple layers and levels of output might exist in a complex LLM application, such as one that employs web search, function calling, tool use, or downstream LLMs.

References

- OWASP AI Exchange on sensitive output handing

Rate limit

While monitoring watches over the number and cadence of interactions, as they can be suspicious, rate limiting should be considered to restrict interactions.

References

- OWASP AI Exchange on rate limiting

Distort input when necessary

Input distortion: Add noise to inputs to reduce sensitivity to adversarial perturbations and reduce triggering of poisoned samples.

References

- OWASP AI Exchange on Input distortion

Minimize attacker information in output

Obscure confidence scores: Limit exposure of model confidence information that could aid attackers.

References

- OWASP AI Exchange on obscuring confidence

AI Deployment in IDEs

IDEs such as VSCode, Windsurf, or Cursor are fully integrated with models or offer LLM integration as a highly desirable option. Although these integrations can significantly increase efficiency, users can inadvertently expose proprietary algorithms, models, API keys, and datasets through AI-powered features. Organizations should explore IDEs with local-only LLM integrations to mitigate risk exposure.

Implement Multilayered Protection/Detection

Although useful, overreliance on system prompts for mitigation of input/output proves suboptimal. System prompts should be thought of as a virtual/temporary/incomplete and tactical mitigation only. Furthermore, system prompts should not be overwritten by user prompts, but should instead require additional layers of guardrails.

Monitoring

Effective monitoring is essential to maintaining AI security over time. AI models and systems must be continuously observed for performance degradation, adversarial attacks, and unauthorized access.

Monitor Interaction/Inference

Interactions with the AI system including API usage and inference should have audit and access logging enabled by default.

Content validation, detection, and filtering (see Model I/O handling) should be complemented with usage and behavior monitoring focused on the interactions themselves. Observing usage for misuse is critical. Anomalous spikes can serve as an effective detection method for abuse.

References

- OWASP AI Exchange on monitoring

Drift Monitoring

Continuously track model performance over time to detect behavioral or data drift. Sudden changes in inference behavior, drift in outputs, or increased refusal rates may indicate adversarial manipulation or unauthorized updates.

References

- OWASP AI Exchange on continuous validation

Output Monitoring

Monitor AI outputs/responses for suspicious patterns. Include pattern-based and behavioral monitoring to identify jailbreak attempts, abuse of multilingual prompts, or bypasses via encoding/compression.

Governance, Risk, Compliance (GRC)

Organizations must align AI initiatives with industry regulations, implement risk-based decision-making processes, and establish frameworks for secure deployment. Continuous testing and evaluation of AI systems are crucial for maintaining integrity, detecting vulnerabilities, and ensuring compliance with evolving standards.

The Biggest Risk of AI Is Not Using AI

It is unrealistic for a security team today to attempt to tell an organization that AI cannot or must not be used. Not only are virtually any controls that a security team might attempt to implement likely to be trivial to bypass, but it is growing more difficult to find any useful enterprise product that does not leverage AI.

To ease stakeholder or GRC concerns, establish an AI GRC board or incorporate AI usage into an existing GRC board. AI-usage policies can be developed to guide users to safe and secure platforms while protecting company data. Although leveraging AI represents risk, the bigger risk is attempting to insist that "AI will not be used here."

References

- OWASP AI Exchange onm AI governance
- OWASP AI Exchange on ISMS for AI

Regularly Test and Tune AI Applications/Models

AI applications and, if possible, the underlying models they employ should be regularly tested to ensure alignment and confirm expected behavior. Though models employed should have been red teamed throughout their development prior to deployment, regular assessments of deployed models and applications should still be performed.

In addition to red teaming, organizations should conduct regular penetration testing of the AI infrastructure, including vector databases, APIs, and connected systems.

References

- OWASP AI Exchange on testing

Model Registries

Model registries are centralized repositories that track and manage ML models through their lifecycle. Benefits include:

- **Access controls** to prevent unauthorized modifications or deployments
- **Monitoring and drift detection** to track performance over time
- **Reproducibility and consistency** ensuring models are deployed with correct configurations
- **Secure storage** of model artifacts and associated metadata
- **CI/CD integration** enabling automated checks during deployment

Account for AI Security and Regulatory Frameworks

The legal and regulatory environment in which AI implementations operate is both complex and rapidly changing. Organizations should track frameworks including: EU AI Act, NIST AI Risk Management Framework, MITRE ATLAS, OWASP AI Exchange, OWASP Top 10 for LLM, and relevant national/regional regulations.

References

- OWASP AI Exchange on compliance

Secure Agentic Systems and AI Autonomy Controls

The rapid advancement of agent-based architectures has introduced new dimensions of functionality, and with them, complex risks. Agentic AI systems are now capable of chaining tasks, invoking functions, retrieving information, and acting independently across platforms.

One of the primary concerns with agentic systems is scope creep. Without explicit constraints, agents may invoke unintended tools, access sensitive data, or generate unpredictable behaviors.

Effective deployment of agentic AI should include:

- **Defined function scope—**Configure agents with clearly delineated permissions. Limit access to only the tools and data required.
- **Execution isolation—**Use sandboxing or containerization to restrict the operating environment.
- **API and functional call controls—**Gate access to external actions via allowlists and validate agent-initiated requests at runtime.
- **Feedback loops and escalation—**Ensure agents include confidence thresholds and fallback mechanisms that escalate uncertain decisions to human operators.
- Prompt injection detection and response.

References

- OWASP AI Exchange on agentic AI

Incident Response and Forensics for AI Systems

Despite best efforts in security architecture, AI systems remain susceptible to compromise. The ability to detect, respond to, and investigate AI-specific threats is critical. Model poisoning, prompt injection, data leakage through output generation, and unauthorized model extraction represent new categories of security events.

Organizations should:

- **Capture audit trails across the stack—**Include prompt inputs, augmentation sources, model outputs, function calls, and tool invocations
- **Monitor for indicators of model tampering—**Watch for sudden changes in inference behavior, drift in outputs, or increased refusal rates
- **Employ detection on prompt and output layers—**Include pattern-based and behavioral monitoring
- **Establish a model integrity baseline—**Use cryptographic hashes, model registries, and periodic validation checks

References

- OWASP AI Exchange on monitoring

Glossary

AI bill of materials (AIBOM)—A detailed record of datasets, models, code, and dependencies used in an AI system, modeled after SBOM, for supply chain visibility and governance

Drift monitoring—A process that continuously tracks model performance over time to detect behavioral or data drift

Inference guardrails—Policy-enforcing filters applied to model inputs and outputs to prevent harmful or unauthorized responses

Large language model (LLM)—A type of generative AI model trained on massive datasets to understand and generate natural language

Model registry—A centralized repository to manage, version, and govern ML models throughout their lifecycle

Multimodal model—An AI system capable of interpreting and generating multiple data types

Prompt injection—An adversarial attack where inputs are crafted to override model instructions or extract confidential information

Retrieval-augmented generation (RAG)—A technique that enhances LLMs by combining them with external vector databases to retrieve relevant context

Transfer attack—An attack developed against one model that is effective against other similar models, even without direct access to the target

Trusted execution environment (TEE)—A secure enclave within a processor that protects sensitive computations and model data

Vector database (vectorDB)—A specialized database that stores data in high-dimensional vector space, enabling fast semantic search

Zero model trust—A security principle recognizing that AI models cannot be fully trusted and require external controls and validation