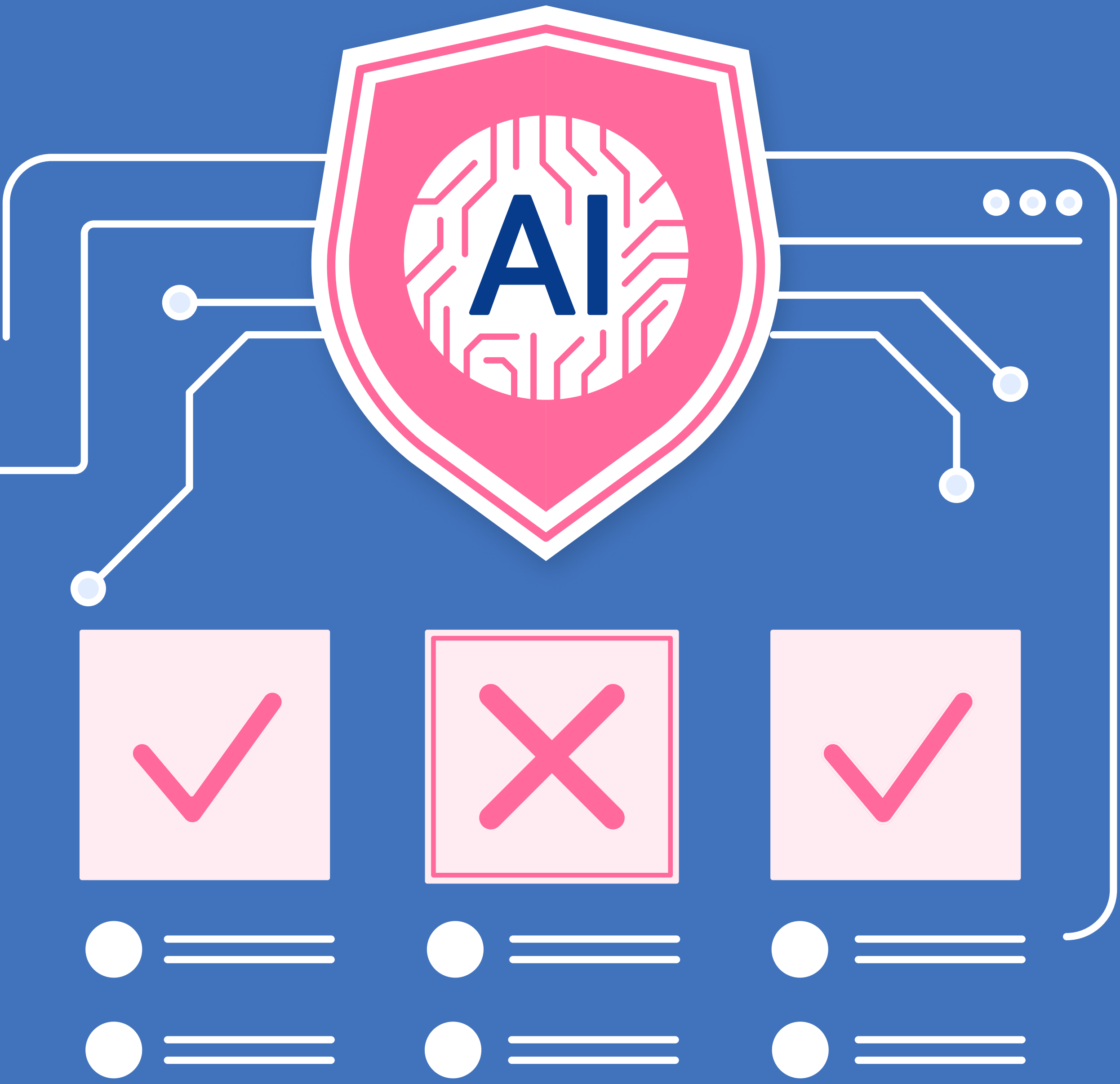
Mend.io

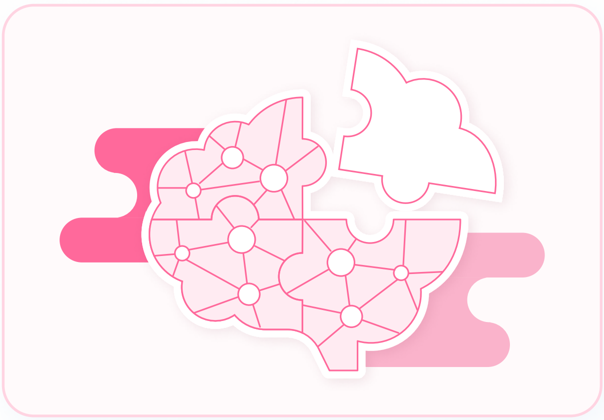
Your AppSec strategy isn’t ready for AI. Yet.

A CISO’s guide to securing AI from the start



Executive summary

The rapid adoption of Artificial Intelligence (AI) in 2025 has transformed how organizations operate, making AI a core component of modern applications. From chatbots enhancing customer interactions to autonomous agents streamlining workflows, AI systems are now integral to the infrastructure of countless industries. However, the security of these systems has not kept pace with their rapid deployment, leaving organizations vulnerable to emerging threats. These include data leakage, where sensitive or proprietary information may be inadvertently exposed; malicious use of AI systems, such as their exploitation for generating convincing phishing attacks or deepfakes; and reputational harm resulting from biases or errors in AI decision-making. Additionally, vulnerabilities in AI models themselves, such as adversarial attacks that manipulate system outputs, further underscore the need for robust security measures tailored to this rapidly evolving technology landscape.



Failing to secure AI applications not only exposes organizations to data breaches, regulatory penalties, and reputational damage but also undermines the trust and confidence of users. As AI continues to revolutionize industries, securing its adoption is essential to ensuring its potential is harnessed responsibly and safely.

Current application security (AppSec) strategies are not fully equipped to address the unique risks posed by AI, particularly those associated with Large Language Models (LLMs). Traditional AppSec tools are designed to scan for known vulnerabilities and mitigate relatively clear-cut risks, such as SQL injection, cross-site scripting (XSS), and insecure configurations. However, AI introduces not only these conventional risks but also a slew of novel challenges that AppSec tools currently lack visibility into. For example, prompt injection attacks exploit the inherent flexibility of AI models, while sensitive data disclosure risks arise from the inadvertent generation or retention of proprietary or user data. Additionally, the concept of excessive agency—where AI systems make decisions or take actions beyond their intended scope—poses complex risks that defy straightforward detection and remediation using existing AppSec frameworks. These gaps highlight the urgent need for security strategies that are specifically tailored to the dynamic and opaque nature of AI systems.

Key Challenges In AI Security

Inherent trust issues

AI systems operate on probabilistic outputs and are vulnerable to manipulation, opaque decision-making, and over-reliance by users. These factors can lead to data breaches, biased results, or costly missteps.

Emerging threats

Novel vulnerabilities such as unbounded consumption, model poisoning, and vector weaknesses pose significant risks to AI applications.

Lagging AppSec strategies

Traditional security approaches often overlook AI's unique risks, leaving gaps in threat modeling, incident response, and governance.

Proactive Security Measures For AI

Champion proactive security by design

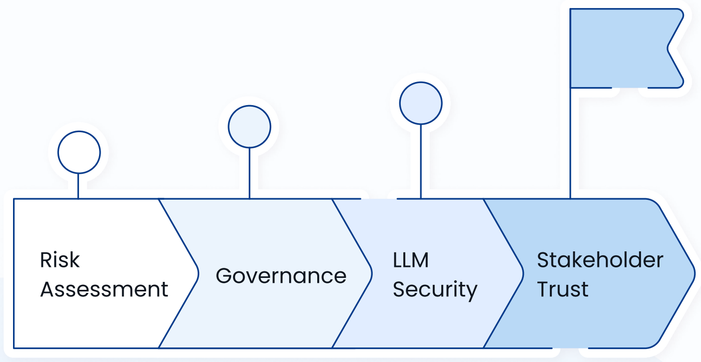
Build AI-driven applications with security embedded from the ground up, rather than relying on retroactive safeguards.

Recognize AI as an untrusted user

Treat LLMs and AI agents as potentially malicious actors, implementing strict input validation, output sanitization, and access control.

Incorporate AI risks into AppSec frameworks

Update security strategies to account for AI-specific vulnerabilities and align with established guidelines like the OWASP LLM Top 10.



This guide equips Chief Security Officers (CSOs) and security professionals with the tools and insights needed to address AI-related risks head-on, providing a roadmap for aligning AI security with broader organizational goals. By prioritizing a proactive, security-first approach to AI, organizations can safeguard their operations and maintain the trust of their stakeholders.

The AI landscape

As organizations increasingly integrate AI into their applications, securing these AI-driven components has become critical. In 2025, AI is ubiquitous, powering everything from business workflows to decision-making systems. Conversational AI, such as OpenAI's ChatGPT and Google's Bard, is frequently embedded within customer-facing applications to enhance service interactions, automate support, and streamline communication. These systems handle billions of user interactions daily, underscoring their deep integration into business-critical applications. For CSOs, understanding how and where AI components, especially conversational AI, are deployed is essential to addressing unique vulnerabilities and ensuring a robust security posture across the organization's digital ecosystem.

Adoption of AI

AI technologies, such as LLMs and autonomous agents, are deeply embedded into modern applications, where they are leveraged to analyze vast datasets, automate complex tasks, and deliver personalized user experiences. From chatbots in customer service to predictive analytics in supply chain management, AI's adoption has revolutionized industries. However, this pervasiveness introduces unique risks that traditional AppSec frameworks may not adequately address.

Trust issues

Unlike conventional software, AI operates on probabilistic outputs and is susceptible to manipulation or misinterpretation. Despite their capabilities, AI systems lack inherent trust mechanisms. Here are just a few key trust-related challenges to be thinking about when it comes to using AI.

Sensitive data processing

AI agents frequently process sensitive information, such as personal user data or proprietary business insights. A breach or misuse of this data could have catastrophic and potentially costly consequences.

Over reliance by end users

Users often place blind trust in AI-generated outputs, assuming them to be accurate or reliable. This over-reliance can lead to poor decision-making, especially when the AI produces incorrect or biased results.

Opaque decision making

The "black-box" nature of many AI models creates a lack of transparency, making it difficult for users to reference the source of the information, understand how decisions are made or identify potential biases.

3 Steps to secure AI across your organization

Securing AI is a critical component of strengthening your organization's overall cybersecurity strategy. The rapid adoption of AI has outpaced the development of comprehensive security strategies, leaving many organizations exposed to novel threats. Here are three key steps to secure AI across your organization's application attack surface.

01

Recognize AI's role in cybersecurity

AI is no longer just an end user tool; it is an integral component of modern applications and systems. Its security must be prioritized alongside current AppSec strategies to secure software and infrastructure.

02

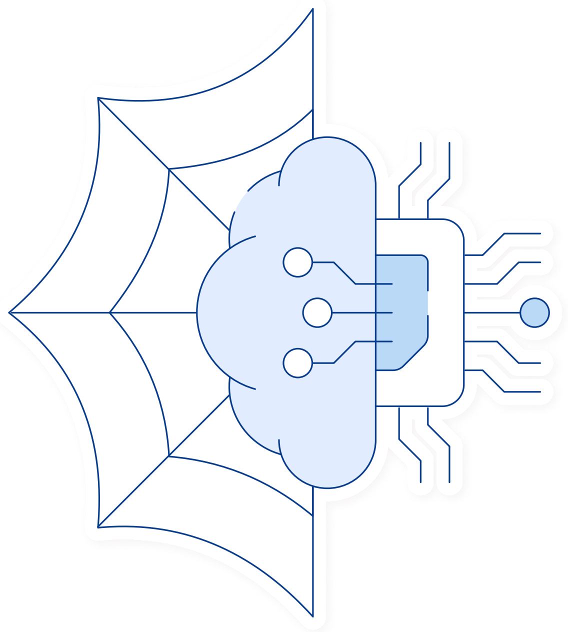
Integrate AI specific risks into AppSec strategies

Current AppSec frameworks often fail to account for AI's unique vulnerabilities. CSOs need to advocate for the inclusion of AI-specific risks in threat modeling, incident response plans, and governance policies.

03

Champion a proactive approach

Waiting for AI-related incidents to occur before taking action is not an option. CSOs must proactively educate their teams, invest in AI security training, and collaborate with developers to ensure security by design. Reading this paper is already putting you steps ahead.



The transformative power of AI comes with significant responsibility —after all, Spider-Man taught us, "with great power comes great responsibility." Organizations that fail to secure their AI applications risk exposing themselves to data breaches, regulatory penalties, and reputational damage. Moreover, the trustworthiness of AI systems directly impacts customer and user confidence and overall business success. By addressing these challenges head-on, CSOs can ensure that AI serves as a driver of innovation rather than a source of vulnerability.

Understanding AI specific risks

Securing AI applications requires a nuanced understanding of their unique vulnerabilities. Traditional AppSec approaches address some general security concerns but often overlook risks inherent to AI. As AI systems evolve and integrate further into business processes, addressing these risks becomes an essential component of any comprehensive security strategy. However, knowing where to start can be challenging. While the OWASP LLM Top 10 may not encompass every potential risk, it provides a solid foundational framework to identify key vulnerabilities, understand the risks you need coverage for, and begin planning an effective security strategy to mitigate threats.

OWASP LLM Top 10: A framework for AI security

The OWASP LLM Top 10 serves as a foundational framework for addressing security concerns specific to AI-driven applications. This evolving guideline highlights key vulnerabilities and provides actionable recommendations for mitigating risks associated with LLMs. For organizations adopting AI, the framework offers a baseline to ensure security considerations are not overlooked in design, deployment, or maintenance.

The latest 2025 list highlights a deeper understanding of risks associated with LLMs and introduces important updates reflecting real-world applications. It expands on previous issues, such as broadening Denial of Service to Unbounded Consumption, addressing resource management, and unexpected costs in large-scale LLM use. New entries, like Vector and Embeddings, provide guidance on securing Retrieval Augmented Generation (RAG) and embedding-based methods, which are now essential practices. System Prompt Leakage addresses vulnerabilities in prompt isolation, emphasizing the need for better safeguards after recent exploits.

OWASP LLM Top 10 Summary

LLM01 2025 Prompt Injection

Definition

Prompt injection occurs when user inputs are crafted to manipulate an LLM's behavior or outputs in unintended ways. This can include bypassing safety protocols, injecting harmful instructions, or causing the model to generate unauthorized or harmful content.

Examples

An attacker manipulates a customer service chatbot to leak confidential information or exploits a model to generate malicious commands by embedding hidden prompts within user inputs.

Potential Consequences

- Data leakage
- Unauthorized actions
- Bypassing of safety controls
- Generation of harmful content
- Loss of trust in the application

LLM02 2025 Sensitive Information Disclosure

Definition

Sensitive information disclosure occurs when LLMs inadvertently expose confidential data, including personal identifiable information (PII), financial records, health details, proprietary algorithms, or other sensitive business information, either through their outputs or by mishandling user inputs.

Examples

A model trained on unfiltered data unintentionally generates sensitive financial details, or an LLM embedded in a business application leaks proprietary algorithms or trade secrets.

Potential Consequences

- Privacy violations
- Data breaches
- Intellectual property theft
- Reputational damage
- Legal liabilities
- Loss of user trust

LLM03 2025 Supply Chain

Definition

Supply chain vulnerabilities in LLMs arise from the reliance on third-party pre-trained models, training data, and deployment platforms, which can be manipulated or compromised to introduce biases, security risks, or failures in the system.

Examples

An attacker tampers with third-party pre-trained models to inject harmful behavior or biases, or compromises a fine-tuning method like LoRA to alter the model's functionality. On-device LLMs also increase risks of unauthorized modifications or tampering during deployment.

Potential Consequences

- Biased or harmful outputs
- System compromise
- Intellectual property theft
- Reduced system reliability
- User harm

LLM04 2025 Data and Model Poisoning

Definition

Data and model poisoning involves manipulating training data, fine-tuning datasets, or embeddings to introduce vulnerabilities, backdoors, biases, or harmful behaviors into an LLM, compromising its security, integrity, or ethical outputs.

Examples

An attacker poisons pre-training data to introduce toxic language patterns, embeds a backdoor that activates harmful behavior with a specific input, or uses malicious pickling to execute harmful code when a model is loaded.

Potential Consequences

- Biased or harmful outputs
- Degraded model performance
- Exploitation of downstream systems
- Increased risk of backdoor activation or malicious actions

LLM05 2025 Improper Output Handling

Definition

Improper output handling occurs when outputs generated by LLMs are insufficiently validated, sanitized, or controlled before being passed to downstream systems, leading to potential security vulnerabilities or unintended consequences.

Examples

LLM-generated content triggers Cross-Site Scripting (XSS) in a web application, allows Server-Side Request Forgery (SSRF) through improperly sanitized URLs, or enables remote code execution due to privilege escalation.

Potential Consequences

- Exploitation of downstream systems
- Including privilege escalation
- Injection attacks (e.g., XSS, SQLi)
- Data breaches
- Unauthorized actions
- Compromise of application integrity

LLM06 2025 Excessive Agency

Definition

Excessive agency arises when LLM-based systems are granted overly broad functionality, permissions, or autonomy, enabling them to perform unintended or harmful actions in response to manipulated, ambiguous, or unexpected outputs.

Examples

An LLM-enabled agent uses excessive permissions to delete files on a system, invokes malicious extensions due to manipulated prompts, or performs unauthorized actions triggered by hallucinated outputs or compromised peer agents in a multi-agent system.

Potential Consequences

- Data breaches
- Unauthorized system modifications
- Denial of service
- Loss of confidentiality, integrity, or availability of connected systems
- Increased vulnerability to exploitation

LLM07 2025 System Prompt Leakage

Definition

System prompt leakage refers to the exposure of the system instructions or prompts used to guide an LLM's behavior, which may inadvertently contain sensitive information or enable other vulnerabilities.

Examples

A system prompt includes connection strings, API keys, or role descriptions that, when leaked, allow attackers to bypass authorization checks or compromise the application. Attackers might infer system guardrails and restrictions by interacting with the model and analyzing its responses.

Potential Consequences

- Unauthorized access to sensitive data
- Bypassing of security mechanisms
- Exposure of application design details
- facilitation of other attacks, such as privilege escalation or unauthorized system access

LLM08 2025 Vector and Embedding Weaknesses

Definition

Vulnerabilities in the generation, storage, or retrieval of vectors and embeddings can be exploited in systems utilizing techniques like Retrieval Augmented Generation (RAG) to manipulate outputs, inject harmful content, or access sensitive information.

Examples

Malicious actors tamper with vector databases to introduce harmful or misleading embeddings, resulting in manipulated model outputs. An attacker reverse-engineers stored embeddings to infer sensitive data or exploits poorly secured vector retrieval mechanisms to gain unauthorized access to critical information.

Potential Consequences

- Compromised data integrity
- Unauthorized access to sensitive information
- Security breaches
- Model manipulation leading to harmful or inaccurate outputsity breaches

LLM09 2025 Misinformation

Definition

Misinformation occurs when LLMs generate false or misleading information that appears credible, often due to hallucinations, biases in training data, or incomplete information.

Examples

An LLM produces a fabricated legal precedent during a court briefing, generates incorrect medical advice in a healthcare application, or provides misleading business data in response to financial inquiries.

Potential Consequences

- Security breaches
- Reputational damage
- Legal liability
- the propagation of false information that can influence critical decisions or processes

LLM10 2025 Unbounded Consumption

Definition

Unbounded Consumption occurs when LLM applications allow users to perform excessive and uncontrolled inferences, leading to resource depletion, economic losses, or service disruption.

Examples

A malicious actor conducts automated requests to overload the system, resulting in denial of service (DoS). Excessive queries incur significant cloud costs, or a competitor clones a model's behavior through repeated inference attempts.

Potential Consequences

- Service outages
- Financial strain due to high operational costs
- Intellectual property theft through model behavior replication
- degraded user experience

Legal risk for AI models and frameworks

Legal risks for LLMs and frameworks stem from several nuanced and complex areas. For example, open source and proprietary licensing agreements can create uncertainty around permissible usage, especially when LLMs are built on datasets or frameworks with diverse sources and varying license terms. If an organization uses an LLM trained on data with restrictive licenses, it could face serious legal repercussions. If the data in training models includes copyrighted data, there may be issues related to intellectual property rights, as even inadvertent reproduction can lead to compliance violations. International regulations governing data usage, privacy, and AI deployment vary widely, making it difficult for organizations to ensure compliance across regions.

Having visibility and coverage into these areas is essential to avoid legal exposure and reputational damage. However, achieving this is challenging due to the opaque nature of AI training datasets, the rapid evolution of licensing and regulatory landscapes, and the technical complexity of tracing how LLMs derive and use their outputs.

Highlighted Risks

01

Violation of open source licenses

Many LLMs and frameworks are built on open source components with strict licensing requirements. Failure to comply with these terms, such as attribution, non-commercial use, or share-alike provisions, can result in legal disputes or loss of rights to use the software.

02

Unauthorized data usage

LLMs trained on datasets containing copyrighted or proprietary material without proper permissions can expose organizations to legal challenges, particularly from content owners.

03

Redistribution and derivative works

Modifying or distributing pre-trained models without adhering to the licensing terms, especially in proprietary or open source hybrid licenses, can breach legal agreements.

04

Export and data residency compliance

LLM frameworks may inadvertently violate international laws, such as export controls or data residency requirements, by deploying or training models on sensitive datasets across borders.

05

Misrepresentation of licensing terms

Inadequate understanding of a framework's licensing terms—such as those requiring royalties, revenue sharing, or restrictions on commercial use—can lead to unintended legal or financial consequences.

AI as a trusted user is a flawed assumption

Integrating LLMs into applications offers significant potential and introduces unique security challenges detailed in the OWASP LLM Top 10. A critical consideration is the architectural positioning of LLMs within your system. While LLMs can significantly enhance application capabilities, it's essential to approach their integration with a security-first mindset. By treating LLMs as untrusted users, proactively designing secure architectures, and adhering to established best practices, organizations can effectively mitigate potential security risks associated with LLMs.

01

Treat LLMs as untrusted users

Designing systems with LLMs at their core assumes inherent trustworthiness. However, it's prudent to treat LLMs as potentially malicious users. This perspective helps in implementing robust control mechanisms and safety-first architectures, mitigating risks associated with LLM outputs. For instance, an LLM might inadvertently generate harmful commands or disclose sensitive information if not properly managed.

02

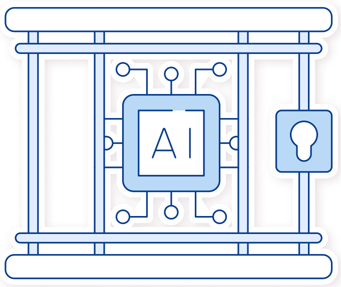
Proactive security design

Relying solely on retrofitting security measures or external safeguards like firewalls is insufficient. Secure LLM applications must be built with proactive security designs from the outset. This includes implementing robust control mechanisms and safety-first architectures to prevent unauthorized access and misuse. For example, without proper safeguards, an LLM could be manipulated through adversarial inputs to perform unintended actions.

03

Adopt and adapt best practices

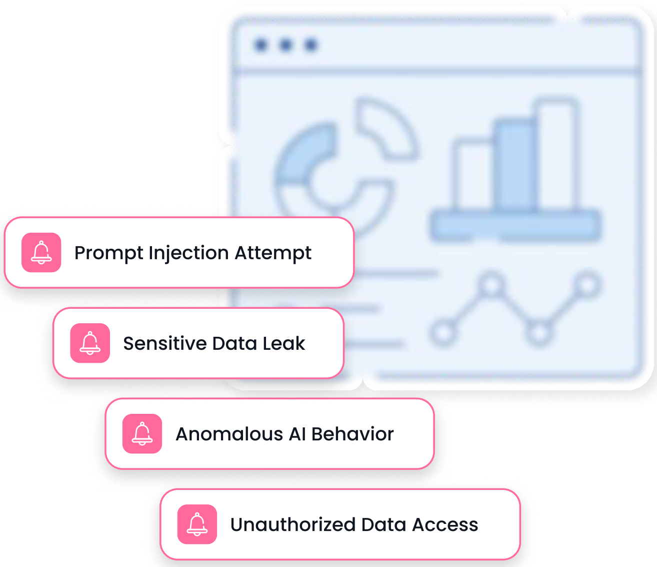
Frameworks such as the OWASP Top 10 LLM Applications & Generative AI provide evolving guidelines to help developers anticipate risks and implement safer LLM-enabled systems. These guidelines cover areas like access risks, data manipulation, and reputational threats, offering a comprehensive approach to AI security. By adhering to these best practices, organizations can enhance the resilience of their AI systems against emerging threats.



Understanding the need to treat LLMs as untrusted users is just the beginning. Once this mindset shift is in place, the next step is translating it into action through security-aware design principles, practical safeguards, and governance strategies. From architecture to implementation, each layer must work together to reduce exposure and maintain control over AI behavior.

Mitigation and governance of AI risks

Addressing AI specific security risks requires a proactive and comprehensive approach that integrates AI into broader security frameworks while accounting for its unique vulnerabilities. Here are some key measures to take to ensure a robust security posture for AI applications.



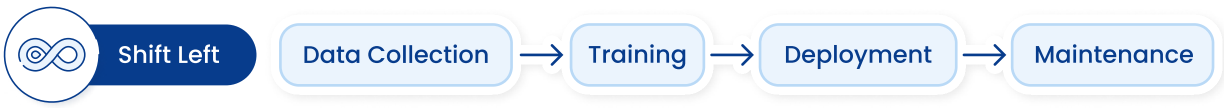
1. Integrate AI into AppSec strategies

AI applications should be treated like any other software product during security assessments. Incorporate AI-specific considerations into existing AppSec tools and methodologies, such as threat modeling and secure coding practices. By leveraging these established tools, organizations can identify and mitigate risks more effectively without reinventing their security processes.

2. Establish AI risk governance

Define clear policies that address the unique risks associated with AI systems. Assign dedicated ownership for AI security within the organization to ensure accountability and consistent management of security concerns. Governance frameworks should include guidelines for evaluating risks, securing data, and ensuring compliance with evolving regulations.

3. Secure AI from the start



Security for AI applications must begin at the design phase. Adopting a "Shift Left" approach encourages collaboration between developers and security teams early in the development lifecycle. Security checkpoints should be integrated into every stage of the AI lifecycle, from data collection and training to deployment and maintenance. This helps identify and address vulnerabilities before they can be exploited.

4. Mitigation strategies for top risks

Treat AI as an untrusted user

LLMs and other AI agents should be treated as untrusted components within the system. Implement strict controls to verify and authorize their outputs before execution.

Technical safeguards

Follow the detailed technical mitigation strategies highlighted in the OWASP LLM Top 10 guide to prevent issues such as prompt injection or data poisoning.

Training and awareness

Educate developers and users about AI-specific security risks, including supply chain vulnerabilities and sensitive information disclosure. Regular training sessions can reduce human error and enhance organizational preparedness.

Red teaming for AI systems

Conduct red-team testing to simulate adversarial attacks on conversational AI systems. This approach helps uncover potential vulnerabilities and improves resilience against real-world threats.

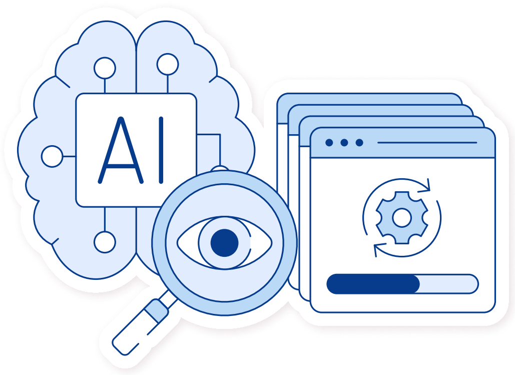
Identify AI model license risks

Organizations must ensure compliance with the licensing terms of AI models and frameworks to avoid legal and financial repercussions. Use a code scanning tool to conduct thorough license reviews, ensuring that usage adheres to open source or proprietary requirements, and maintaining documentation of all model and framework licenses.

5. Adopt frameworks and guidelines

Leverage existing frameworks, such as the OWASP LLM Top 10, to help your teams understand and address known vulnerabilities like improper output handling, excessive agency, and misinformation. Continuously adapt security protocols to align with evolving best practices and emerging risks. Ensure AI-specific guidelines are fully integrated into the organization's broader AppSec strategies.

6. Continuous monitoring and auditing



Automate monitoring tools to track AI systems for unusual behaviors, such as unbounded consumption or unauthorized data access. Conduct regular audits of the AI lifecycle, including data integrity checks, model updates, and permission reviews, to promptly address new risks.

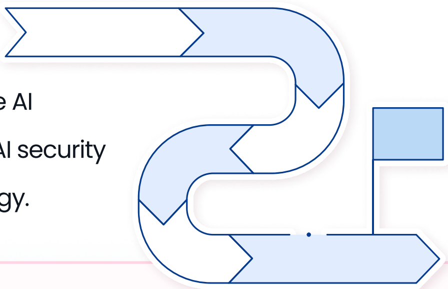
7. Develop incident response plans

Prepare for AI-specific incidents by establishing a response plan tailored to risks like prompt injections or misuse of AI capabilities. Regularly test these plans through simulations to ensure the organization can respond quickly and effectively during a security event.

By integrating these measures into AppSec strategies and fostering a culture of security-by-design, organizations can proactively address the unique challenges AI systems pose. Treating AI as an untrusted user, establishing governance frameworks, and adopting technical safeguards are critical steps in building resilient AI applications. As AI becomes increasingly integral to business operations, embedding security into its foundation is not optional—it is essential.

Next steps for CSOs

Chief Security Officers are pivotal in guiding their organizations to secure AI systems effectively. By taking a structured approach, CSOs can ensure AI security becomes an integrated component of the broader cybersecurity strategy.



Assessment and visibility

- Conduct a **gap analysis** to evaluate the organization's current AI security practices against industry standards and best practices.
- Develop an **inventory of AI applications and systems**, identifying how they are integrated into business processes and where vulnerabilities may exist.

Action plan

- **Prioritize risks** based on their potential impact and likelihood, addressing critical vulnerabilities first.
- Implement **quick wins** such as securing data inputs, reviewing output validation processes, and ensuring compliance with AI model license terms.

Build the team

- **Define roles and responsibilities** for AI security, ensuring clear ownership across teams.
- Allocate the necessary **budget and resources** to support AI-specific risk management efforts, including training, tools, and personnel.

Measure progress

- Establish **Key Performance Indicators (KPIs)** to track improvements in AI security, such as reductions in incidents, faster response times, or enhanced compliance rates.
- Regularly **review and update AI security policies** to reflect evolving risks, emerging technologies, and industry developments.

By following these steps, CSOs can establish a strong foundation for mitigating AI-specific risks and ensuring their organizations are prepared to handle the challenges of securing advanced AI systems.

Mend.io

Find out more about securing AI powered applications in your organization.

Book a demo