ONA

REPORT

Securing AI-native software development in 2026

Securing AI-native software development in 2026

AI is moving from simple coding assistants to autonomous agents that build features, refactor codebases, and integrate across systems. This evolution creates a threat landscape that traditional security controls cannot address.

This report examines:

→ The principal security vulnerabilities unique to AI-assisted development.

→ The major regulatory frameworks that now shape AI governance in the EU, US, and UK.

→ A baseline set of security controls required for safe adoption across jurisdictions.

For security leaders, the objective is clear: adapt development security architectures to an agent-driven environment before 2026, when AI integration will be ubiquitous and attack surfaces will be larger, faster-changing, and less transparent.

The new AI software engineering reality 04

Major vulnerability categories of AI in software 05

Regulatory landscape for AI development 07

Baseline security controls for secure development 11

Your 2026 AI security mandate 13

The new AI software engineering reality

By 2026, AI will not just accelerate development workflows but will restructure them.

Coding assistants are now capable of producing entire features, and performing large-scale refactoring, and operating across environments.

Human engineers are shifting towards architectural design and system boundaries, setting quality standards, and ensuring outputs meet security, compliance, and business objectives. The impacts are wide-reaching:

→ **Product and design:** Prototypes and iteration create faster cycles.
→ **Unblocking subject matter experts:** More stakeholders can participate in software.
→ **Automation of toil:** Enforcing standards and remediate vulnerabilities.

As are the risks:

→ **Expanded attack surface:** AI has growing wide-ranging access to repositories
→ **Reduced visibility:** 'Black box' decisions complicate bug and incident investigations.
→ **Skills erosion:** Over-reliance on AI risks degrading core engineering expertise.
→ **Ethical exposure:** AI-generated code can embed bias and infringe IP.

Major vulnerability categories of AI in software

The following categories represent the most significant threats we've identified through our research and real-world incident analysis:

Source code and IP exposure

Developers can unintentionally disclose proprietary code or algorithms by pasting them into external models. Conversely, AI tools may suggest code fragments from their training data, creating both IP liability and security risk.

Malicious code injection via AI assistants

Attackers can craft prompts that cause assistants to insert backdoors. As AI-generated output grows, manual review capacity falls. The ‘smoke and mirrors’ attack encodes malicious intent into benign-looking artefacts (i.e., commit histories, comments) with high success rates.

Poisoned development models

Contamination of AI models can happen at ‘source’ in the models that are being used for code generation. A model can suggest insecure code snippets that evade both human review and automated tools. The CODEBREAKER attack, demonstrated at USENIX Security 2024, shows how adversaries can fine-tune LLMs on code samples to insert hidden vulnerabilities.

CI/CD pipeline exploitation

Agents with CI/CD access can be manipulated to bypass tests, alter build configurations, or insert malicious payloads. The 2025 Amazon Q VS Code extension incident that was triggered by a misconfigured GitHub token demonstrated how such compromises propagate.

A threat actor exploited a misconfigured GitHub token in the project's build process to inject malicious code into the extension's repository. The poisoned update (v1.84.0) was published before AWS caught it, and only a syntax error prevented the payload from executing.

Shadow development tools

Use of unauthorized AI tools bypasses governance and may transmit sensitive code to insecure services leaking IP or sensitive source code.

Compromised code review and testing

As code volume increases, organizations turn to AI for review and scanning, but these scanning systems themselves can be evaded. Studies show that even advanced code generation models fail to flag obvious vulnerabilities unless explicitly prompted.

Regulatory landscape for AI development

We also examined four major regulatory approaches that are shaping the future of AI governance in software development. The EU AI Act, the United States' NIST AI Risk Management Framework, the global ISO/IEC 42001:2023 standard for AI management systems and the UK's principles-based approach.

EU AI Act

A risk-based, binding regulation with phased implementation:

→ **Feb 2025:** Ban on ‘unacceptable risk’ systems (e.g., workplace emotion recognition, social scoring). Mandatory AI literacy programs.

→ **Aug 2025:** General-purpose model providers must publish technical documentation, training data summaries, and copyright compliance processes.

→ **Aug 2026:** High-risk AI in hiring, credit, or critical infrastructure must meet full quality management, dataset, and human oversight requirements.

The Act has faced a lot of criticism. In July 2025, 45 leading European companies, including Mistral AI's CEO Arthur Mensch, signed an open letter pleading for a two-year 'clock stop' on implementation. They argue the Act's complexity threatens to strangle European AI innovation in its crib. Meta went further, refusing to sign entirely, declaring it 'an overreach that will throttle frontier AI development in Europe.' Even Google, while agreeing to sign, warned the rules risk 'slowing Europe's development and deployment of AI.'

NIST AI Risk Management Framework

The United States has taken a different path by developing a voluntary framework emphasizing flexibility and self-regulation. It organizes around four core functions that organizations are encouraged to implement:

1. **Govern** — Roles, policies, accountability.
2. **Map** — System context, capabilities, stakeholders, risks.
3. **Measure** — Trustworthiness metrics, pre-deployment testing.
4. **Manage** — Risk prioritization, controls, incident response.

Critics argue that its high-level nature provides little practical guidance—terms like 'trustworthy AI' and 'appropriate risk management' mean different things to different organizations. However, unlike the EU's prescriptive approach, the NIST framework allows organizations to tailor their risk management to specific contexts and capabilities. The framework has gained traction through market pressure with contracts now requiring 'NIST AI RMF compliance' even though the framework itself does not mandate.

ISO/IEC 42001:2023

ISO/IEC 42001 attempts to bridge the gap between prescriptive regulation and voluntary frameworks through a certifiable but non-mandatory approach. The standard follows ISO's established methodology, using the Plan-Do-Check-Act cycle familiar to organizations already certified under other ISO standards. Companies seeking certification must establish an AI Management System (AIMS) that addresses the unique challenges AI poses—ethical considerations, transparency requirements, and the need for continuous learning.

The standard specifies 38 distinct controls organizations must implement, covering everything from data governance and model development to third-party management and performance monitoring. These controls are specific and auditable.

UK pro-innovation approach

The United Kingdom has deliberately charted a course between American voluntarism and European prescriptivism, developing what it calls a "pro-innovation" approach relying on existing regulations rather than new legislation. Instead of creating AI-specific laws, the UK has tasked regulators with interpreting and applying five cross-sectoral principles within their domains:

1. **Safety, security and robustness**
2. **Transparency and explainability**
3. **Fairness**
4. **Accountability and governance**
5. **Contestability and redress**

This approach means concrete requirements vary dramatically by sector. A bank using AI for credit decisions faces different obligations than a hospital using diagnostic AI, even though both operate under the same principles.

Critics argue the UK approach creates dangerous regulatory gaps. Without binding requirements, companies can interpret the principles however suits their interests. The reliance on sector-specific regulators risks creating a patchwork of conflicting requirements as each interprets the principles differently. The ICO's November 2024 findings that recruitment AI tools were processing data unfairly and filtering candidates based on protected characteristics illustrates how voluntary principles fail to prevent discrimination.

Baseline security controls for secure development

Regardless of jurisdiction, the following controls are considered foundational for AI-assisted development security.

Identity, access, and secret management

Apply zero-trust principles to AI agents. Treat each agent as a unique identity with least-privilege, revocable access, and full audit logging. This would be in compliance with the EU's audit trail requirements and ISO 42001's control requirements for access management.

Audit logging

Capture prompts, responses, decisions, and model versions in append-only, cryptographically verifiable logs to enable forensic reconstruction and detection of malicious prompt injections.

Sandboxing of AI development environments

Run AI in isolated, ephemeral environments with restrictions to prevent persistence and lateral movement. ISO 42001 mandates controlled environments for development and testing which necessitates sandboxes with strong isolation using containerization or virtualization to limit CPU, memory, and network access. Network segmentation with strict firewall rules prevents lateral movement if an AI system is compromised through any of the attack vectors we've identified.

AI gateway and cost management

Route all model requests through a centralized gateway to enforce policies and usage of LLMs,as well as gain visibility to monitor usage and costs for attribution.

Your 2026 AI security mandate

By 2026, AI will be embedded at every stage of software delivery. While regulation can enforce governance, it cannot replace technical controls. Securing AI-assisted development requires an architecture designed for agents: zero-trust access, immutable logging, sandboxing, continuous scanning, and behavioral guardrails. Organizations that implement these measures now will be positioned to scale AI use without sacrificing security, compliance, or velocity.

Those that delay risk building on foundations they will struggle to protect.

Appendix

Li et al., "LLMs Caught in the Crossfire: Malware Requests and Jailbreak Challenges" (ACL 2025) - MalwareBench study showing 40% compliance rate with malicious requests

Wu et al., "Smoke and Mirrors: Jailbreaking LLM-based Code Generation via Implicit Malicious Prompts" (2025) - 80% success rate with hidden malicious intent

Yan et al., "An LLM-Assisted Easy-to-Trigger Backdoor Attack on Code Completion Models: Injecting Disguised Vulnerabilities against Strong Detection" (USENIX Security 2024)

AWS Security Bulletin AWS-2025-015, "Security Update for Amazon Q Developer Extension for Visual Studio Code (Version #1.84)"

Krebs on Security, "xAI Dev Leaks API Key for Private SpaceX, Tesla LLMs" (May 2025)

Lupin & Holmes, "We hacked Google's A.I Gemini and leaked its source code (at least some part)" (March 2025)

Sajadi et al., "Do LLMs Consider Security? An Empirical Study on Responses to Programming Questions" (Empirical Software Engineering 2025)

The Register, "AI bots hallucinate software packages and devs download them" (March 2024) - Alibaba GraphTranslator incident

Your mission control for AI-native software development.

→ Try us free today: ONA.COM