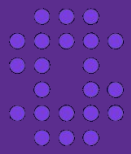


SALT

RESEARCH REPORT

AI Coding Assistants and the New Security Challenge

How organisations are balancing developer productivity with governance, visibility and control

2026 | Salt Security Research

Executive Summary

AI coding assistants have moved beyond the pilot phase. They are embedded in development teams across industries, reshaping how software is built, how fast it ships, and how risk is introduced at scale.

But adoption has outpaced governance. New research reveals that nine in ten security leaders carry concerns about AI-generated code, yet many organisations still rely on the same manual review processes designed for a pre-AI world.

The result is a widening gap between engineering velocity and security control. Closing that gap requires more than policy documents and periodic code review. It requires a fundamental rethink of how governance is embedded into the AI-assisted development lifecycle.

KEY FINDINGS AT A GLANCE

- 67% of organisations report AI coding assistants are now widely adopted across development teams
- 9 in 10 security leaders have active concerns about AI-generated code
- 29% identify insecure coding patterns as the leading risk introduced by AI assistants
- 38% of organisations still rely primarily on manual review for AI-generated code
- 15% cite misalignment with internal security policies as a major concern

AI-Assisted Development Has Already Crossed the Threshold

More than two thirds of organisations surveyed confirm that AI coding assistants are already widely used across development teams. From code completion to debugging, documentation and test generation, these tools now touch nearly every phase of the software development lifecycle.

The productivity gains are real and measurable. Developers generate boilerplate faster, reduce time on routine tasks and ship code at higher velocity. That is precisely why adoption has been so rapid, and precisely why the governance challenge is so urgent.

AI-generated code introduces complexity that traditional security controls were not built to handle. Code can be produced at machine speed, at machine volume, and with machine consistency, including consistent reproduction of insecure patterns. The scale of the problem is structurally different from anything security teams have managed before.

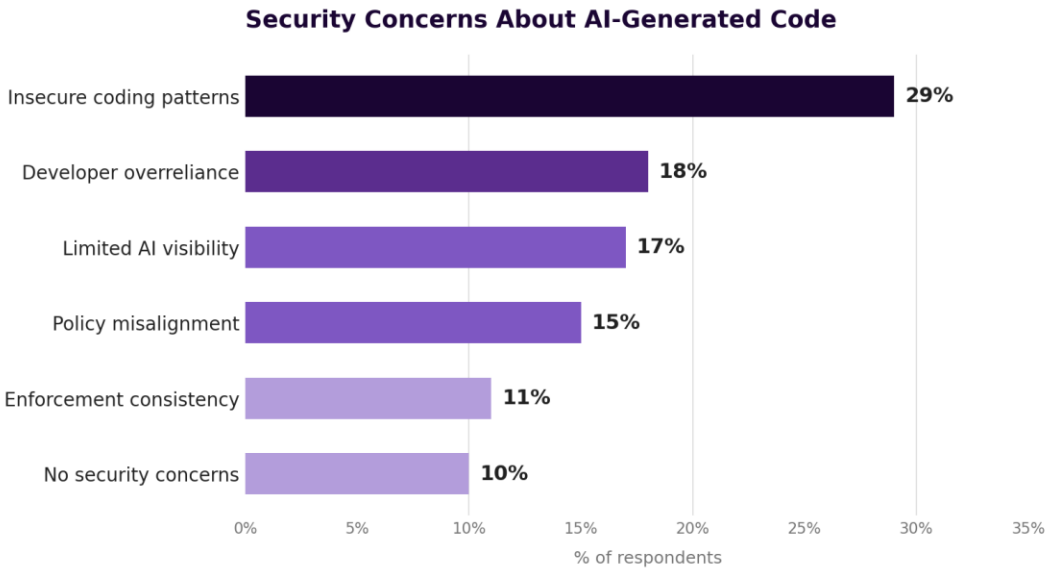
International guidance from agencies including the UK National Cyber Security Centre, CISA and the NSA has reinforced this concern directly. Autonomous and AI-driven systems expand attack surfaces, complicate oversight and erode accountability across software environments. The findings from this research suggest organisations are already encountering these conditions inside their own development pipelines.

Security Concerns Are No Longer a Minority View

Only 10 percent of respondents report having no security concerns about AI-generated code. That figure should stop security leaders in their tracks.

The remaining 90 percent identified at least one material risk. The leading concern, cited by 29 percent of respondents, is the introduction of insecure coding patterns. AI models trained on large codebases can reproduce insecure practices from that training data. They can recommend outdated approaches. They can generate code that appears functional while failing to meet security standards that a human reviewer would catch.

Other concerns reflect the operational realities of AI governance at scale:



Taken together, these findings paint a picture of organisations that recognise the risk but have not yet built the infrastructure to address it systematically. Concern is widespread. Action is inconsistent.

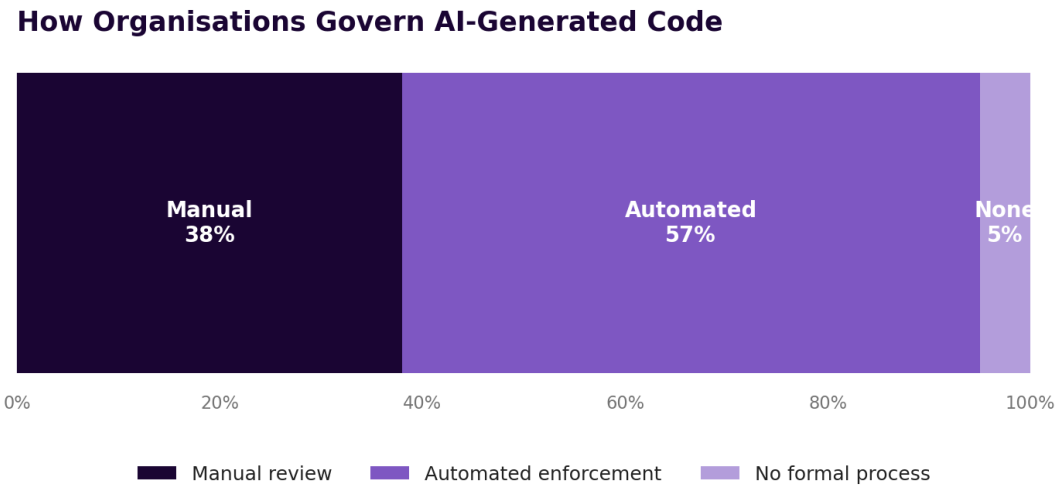
Manual Review Cannot Scale to Machine-Speed Development

One of the most consequential findings in this research is how many organisations are still depending on human-led validation to govern AI-generated code.

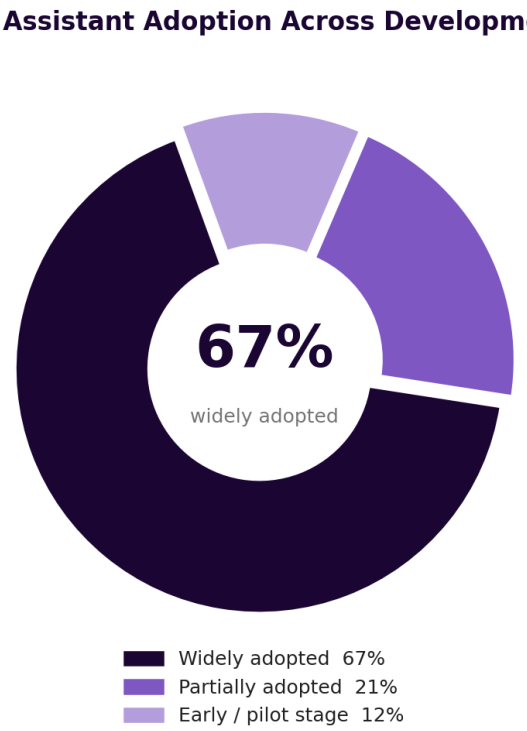
Thirty-eight percent of respondents say they rely primarily on manual review processes. Fifty-seven percent report using automated enforcement mechanisms. The implication is a hybrid model where AI dramatically accelerates code generation, but security assurance still leans heavily on developer judgment and post-generation review.

That model was not designed for this environment. Manual review introduces its own risks:

- Inconsistency across teams and reviewers
- Reviewer fatigue as AI-generated code volumes increase
- Variable interpretation of internal standards
- Scalability ceilings that AI adoption will rapidly exceed
- Gaps between what policies prescribe and what developers actually check



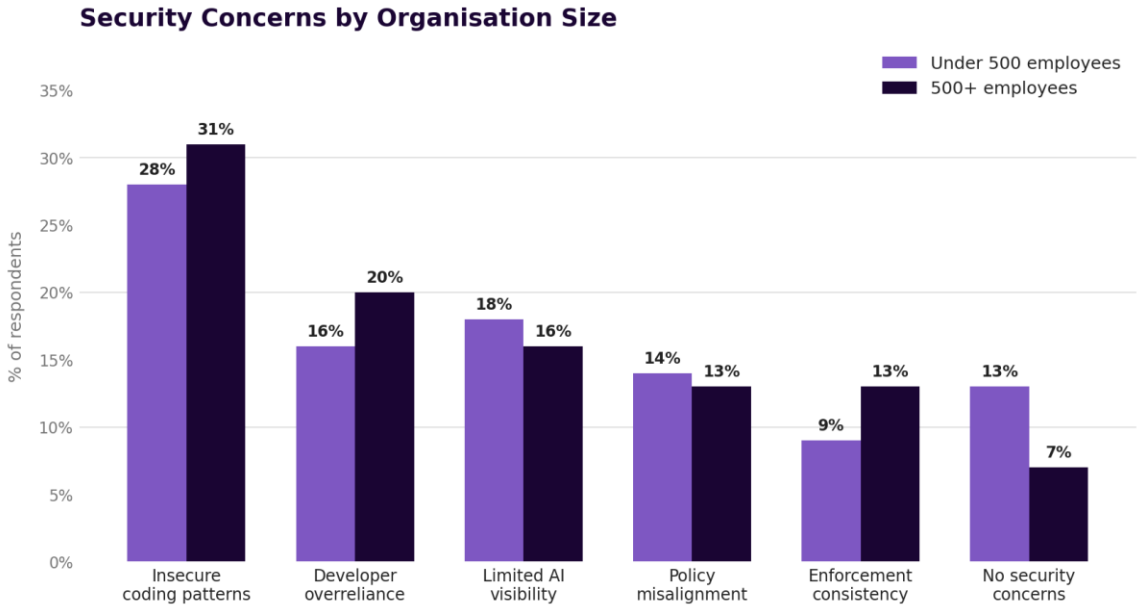
As AI-generated code volumes grow, the burden on human reviewers grows proportionally. Organisations that do not close this gap now will find it significantly harder to close later.



Enterprise Scale Amplifies Every Governance Gap

The governance challenge does not affect all organisations equally. Larger enterprises face a structurally harder problem.

In smaller organisations, concerns centre primarily on insecure coding practices and the absence of formal controls. Those are real challenges, but they are relatively bounded. In larger enterprises, the challenge becomes operational: how do you standardise governance across distributed teams, multiple development environments and heterogeneous tooling at scale?



The data reflects this divide. Enterprises with more than 500 employees are more likely to report concerns around compliance enforcement and overreliance by less experienced developers, while smaller organisations place greater emphasis on code security and visibility. Larger organisations are also less likely to report having no concerns at all.

As AI-assisted development expands inside the enterprise, maintaining alignment between engineering velocity and governance standards becomes increasingly difficult using manual approaches alone.

Five Priorities for Security Leaders

1. Move Beyond Policy Documents

Written guidance is necessary but not sufficient. Policies are only effective if they can be applied consistently at scale, across every team, tool and workflow where AI-generated code appears. The gap between what a policy document says and what developers actually do is where most governance failures originate.

2. Build Visibility Into the AI Layer

Security teams need answers to questions they currently cannot answer: Where is AI-generated code being used? Which tools are producing outputs? How does that code align with internal standards? Without visibility into the AI layer of the development lifecycle, governance becomes guesswork. Visibility is also foundational for compliance, audit and incident response.

3. Reduce Structural Dependence on Manual Review

Human review retains value for high-risk, high-context decisions. But it cannot be the primary control for AI-generated code at scale. Organisations need governance that is proactive, embedded and automated, positioned upstream in the development process rather than downstream in a review queue.

4. Standardise Secure Development Practices Across Teams

Consistency is the foundation of effective governance. Variability between teams, regions and individual developer practices creates the conditions for security drift. The more standardised development processes become, the easier it is to enforce standards, detect deviations and reduce exposure over time.

5. Govern AI as Part of the Software Supply Chain

AI coding assistants are not productivity tools that happen to produce code. They are components of the software supply chain. They should be governed accordingly: with the same rigour, the same visibility requirements and the same accountability structures applied to any other third-party input that touches production software.

International cyber security guidance has consistently made this point. AI security is not a standalone discipline. It should be integrated into existing security and governance frameworks as a first-class concern, not treated as an edge case or a future problem.

The Window for Easy Action Is Closing

AI coding assistants are not going away. The productivity benefits are too significant and adoption is already too widespread for organisations to reverse course. The question is no longer whether AI-generated code will become part of the development lifecycle. In most organisations, it already has.

The next phase of AI-assisted development will be defined by operational maturity. The organisations that build governance infrastructure now, before AI code volumes make the problem unmanageable, will be in a fundamentally stronger position than those that wait.

That means improving visibility into the AI layer. Strengthening governance frameworks. Standardising secure coding practices. Reducing dependence on inconsistent manual review. And treating AI-generated code with the same security rigour applied to any other input that touches critical systems.

The challenge is not whether to govern AI-generated code. The challenge is building the infrastructure to do it at the speed and scale AI demands.

Salt Security helps organisations meet that challenge by providing the visibility, governance and enforcement capabilities needed to secure AI-assisted development at enterprise scale.

Methodology

The research was conducted by Censuswide, among a sample of 100 IT Security Leaders across the UK and USA (Aged 18+). The data was collected between 12.05.2026 - 15.05.2026. Censuswide is a member of the Market Research Society (MRS) and the British Polling Council (BPC), and a signatory of the Global Data Quality Pledge. We adhere to the MRS Code of Conduct and ESOMAR principles

See Salt for yourself

The Salt Agentic Security Platform gives you complete visibility into your entire agentic landscape in less than 10 minutes. See exactly what is exposed, what is at risk, and where to act first.

REQUEST A DEMO

SALT © 2026 Salt Security. All rights reserved. This document is for informational purposes only. Product capabilities are subject to change.

Agentic Security Platform • 2026 • salt.security