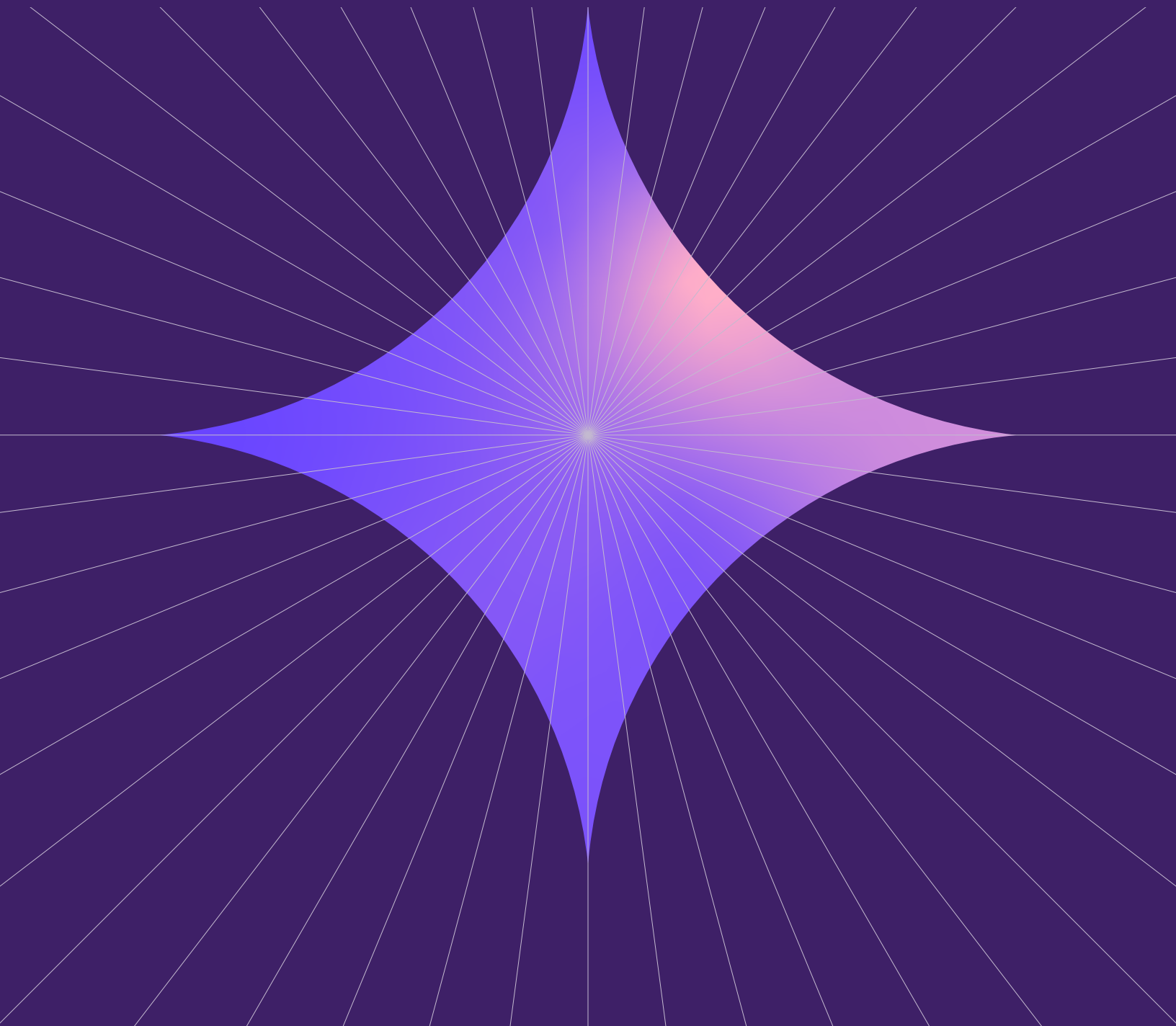
AppSec in the Age of AI

LEGIT

More risks, more opportunities



AI is not only revolutionizing how software gets built, it's also transforming the makeup of software itself, and radically altering what it means to secure software.

What is changing, and how should security evolve? And beyond all the fear and uncertainty, what are the opportunities? How can security take advantage of these changes to solve some of the long-standing AppSec problems and make the process more efficient and effective?

Ultimately, AI is, and should be thought of as, a mix of new cybersecurity risk and new opportunity. AI is adding new levels of complexity and attack surfaces to development, but it's also giving us powerful new ways to manage that risk more effectively. In other words, AI is the problem, but also the solution.

The Changing Software Development Process and Product

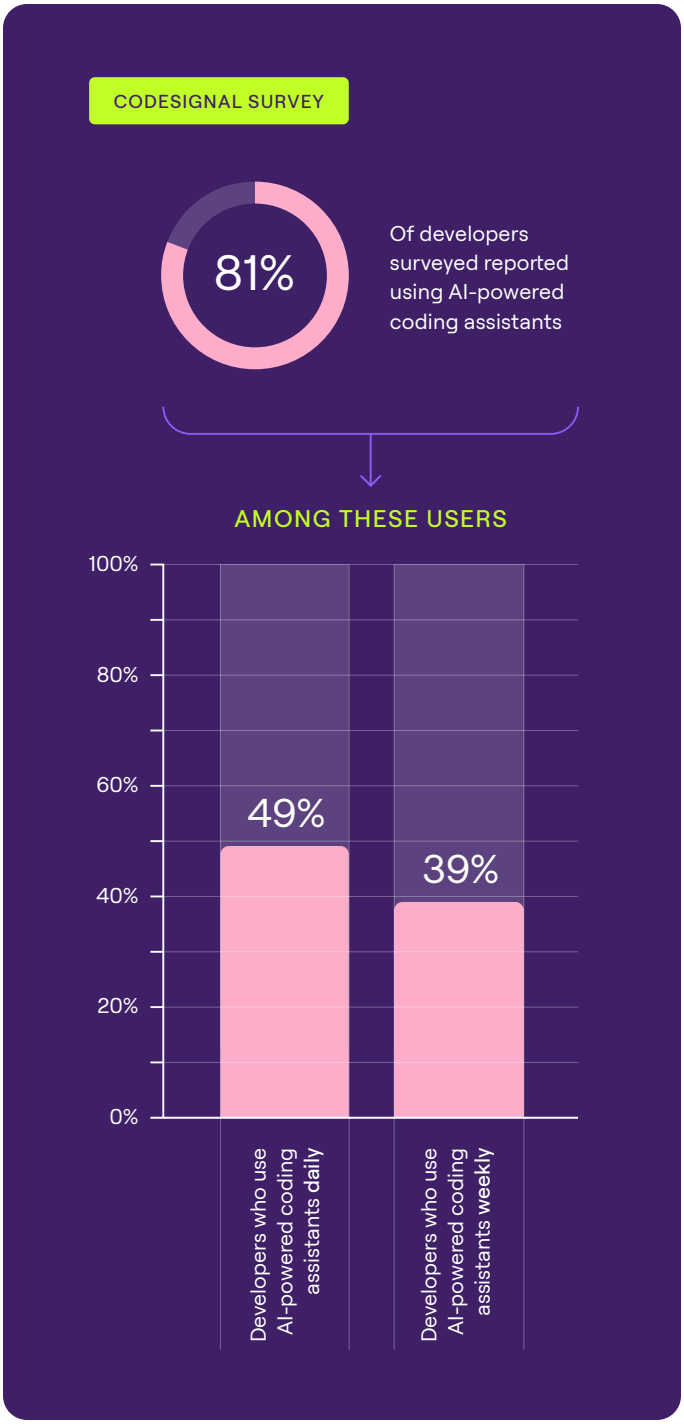
AI changes software development

We're currently experiencing the rise of the AI developer. Almost like pair programming, developers now have another "developer" working alongside them, completing the code and offering suggestions. In turn, the role of the modern software developer is evolving, from that of coder to that of an architect with a "supervisor" mindset.

Here are the ways it's happening:

AI code generating tools

By offering suggestions, code snippets, and writing complete features and pushing them to production, code assistants greatly increase the speed of software delivery and put the engineer in the director's chair as code is developed. A recent CodeSignal survey found that 81% of developers surveyed reported using AI-powered coding assistants. Among these users, 49% utilize them daily, and 39% use them weekly.



In a recent survey of 400 security professionals and developers conducted by Legit, 96% of security and software development professionals report that their companies use GenAI-based solutions for building or delivering applications. 88% of developers surveyed report using AI coding assistants.

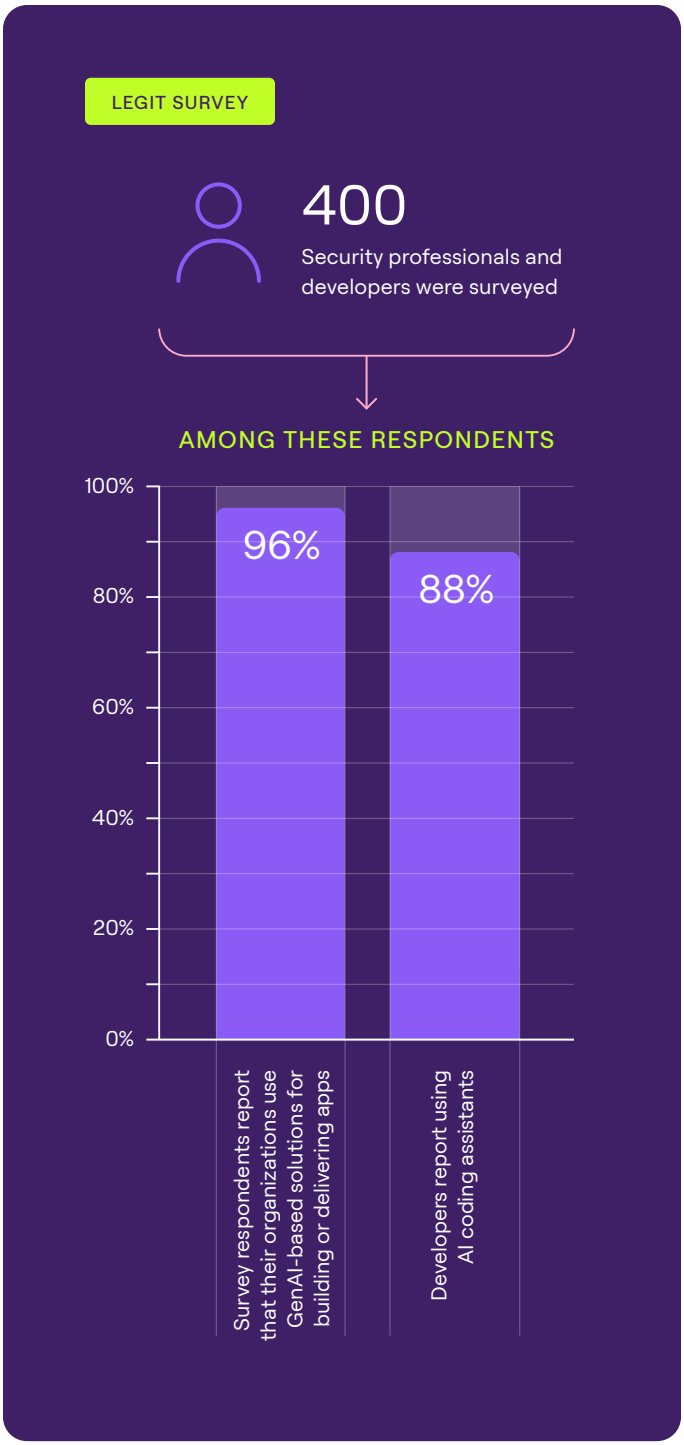
Vibe coding

A term created in February 2025 by computer scientist Andrej Karpathy, vibe coding and vibe coding tools such as Cursor allow developers to use natural language to deliver prompts that drive AI to generate the code itself and perform developer tasks such as running tests. Gartner reports that, by 2028, 40% of new enterprise production software will be created with vibe coding techniques and tools (Gartner, "Why Vibe Coding Needs to be Taken Seriously," May 20, 2025).

LLM agents for coding workflows

"Teams" of AI agents can now work together to plan, write, test, and document code. Each agent uses AI to execute its specific task and then integrates with others on the "agent team" to further the code development process.

At Microsoft's Build 2025 developer conference, CTO Kevin Scott announced that the daily active usage of AI agents has more than doubled compared to the previous year. This surge underscores the rapid integration of AI agents into development workflows.

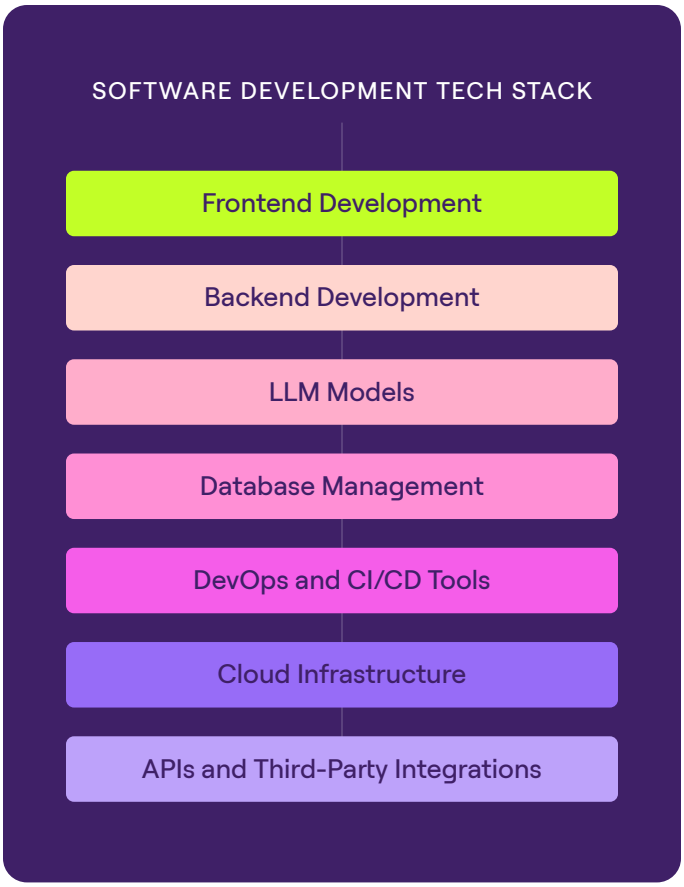


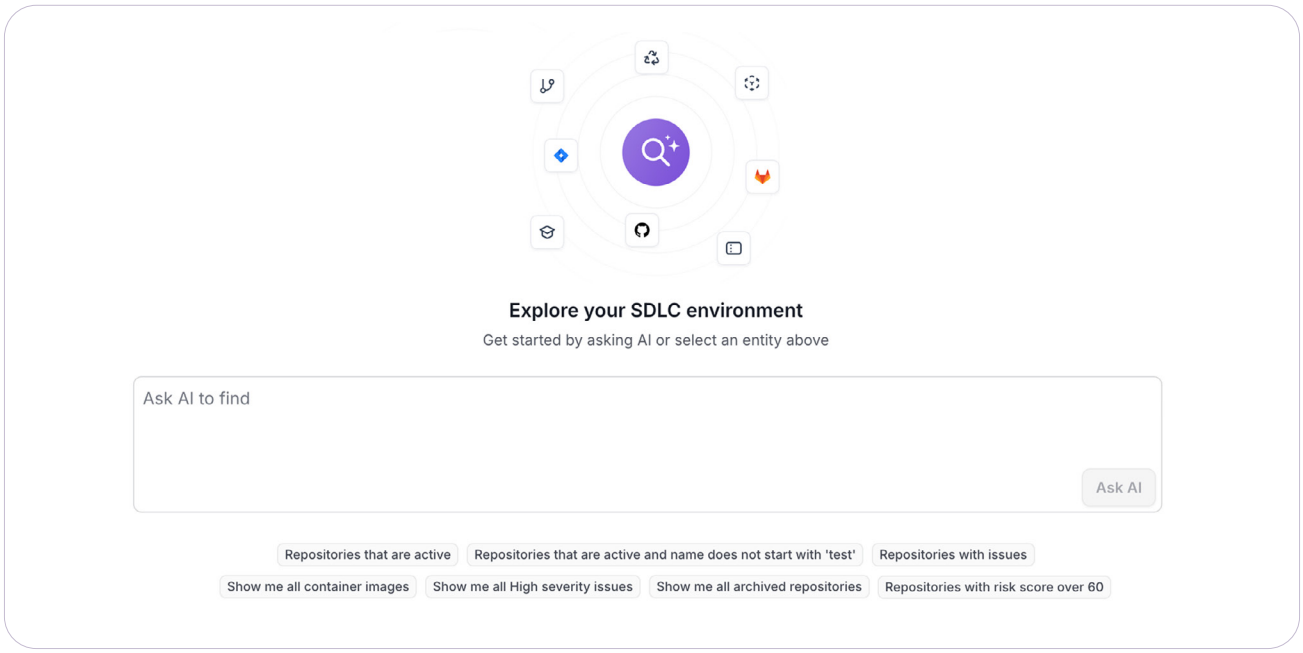
AI changes software itself

The ability to develop software faster and with less coding skills means more people will be creating more software. Not only will there be more of it, but it will also be different.

AI is now a part of the application architecture. Models, LLMs, and agents are key features of the software stack, meaning every software product today has some AI component that is fetching data or communicating with other tools. For instance, the user interface is now a chat, or there is an agent that can query data for you.

And AI is not just enhancing your software, it's another user of your software. As AI agents communicate with each other, every organization should expect their own and outside AI agents to be interacting with their data.





Legit Ask AI Interface

The Risks of AI-Generated Software Development

The bottom line is that AI means more code, a lot more code. And more code that is not at the quality level of human-generated code. Why? For one, because it's trained on code from across the Internet, not all of it high-quality or secure. In addition, AI generators could, and have, created risks like data exposure, supply chain security issues, or the introduction of outdated, vulnerable, or malicious libraries.

At the same time, while the number of lines of code has and is exploding, the level of human capacity for oversight and review has stayed the same. So we've got more code, with more problems, and less oversight. In addition, the nature of AI-generated code itself introduces risk. The problem isn't just that AI is "new." It's that AI models — and the ecosystems they live in — behave fundamentally differently than traditional code and services.

A few key reasons AI introduces fresh risk:

Opaque Behavior

Unlike traditional software, AI outputs aren't always predictable. Attackers can manipulate inputs (prompt injection) or outputs (data poisoning) in ways traditional security models can't catch easily.

Expanded Supply Chains

Organizations now consume pre-trained models, APIs, and AI services from third parties, without always validating their provenance, security practices, or hidden risks.

Secret Leakage and IP Exposure

AI systems trained on internal or external datasets can inadvertently leak sensitive data, secrets, or intellectual property, either through direct queries or indirect model behaviors.

Shadow AI and Rogue Deployments

Teams are integrating AI models and APIs outside of centralized governance. If you thought shadow IT was bad, wait until you see shadow AI.

And AI's attack surface isn't just theoretical anymore. We're seeing early examples of model manipulation attacks, prompt injections into AI-driven apps, poisoned training data, and malicious model updates starting to crop up, and it's only going to get worse.

On "vibe coding," in which the AI prompts are 100% natural language, Gartner recommends companies "limit it to a controlled, safe sandbox for execution. Do not use vibe-coded software in your production efforts until the tools mature further." Further, Gartner said: "The risks are substantial if developers dive in unprepared or use these tools independently." (Gartner, "Why Vibe Coding Needs to be Taken Seriously," May 20, 2025)

Gartner also notes that "By 2027, at least 30% of application security exposures will result from usage of vibe coding practices." (Gartner, "Hype Cycle for Application Security, 2025," July 22, 2025)

And the risk is not just in the code, the tools and services used in AI code generation bring risk into your environment as well.

AI INTRODUCES NEW RISKS

- **Prompt Injection**
  Harmful inputs trick LLMs into making unsafe or unauthorized MCP tool calls. Gartner recently stated that, "Through 2029, over 50% of successful cybersecurity attacks against AI agents will exploit access control issues, using direct or indirect prompt injection as an attack vector." (Gartner, "Hype Cycle for Application Security, 2025," July 22, 2025)
- **Tool Description Poisoning**
  Malicious instructions are embedded in tool metadata or mutate post-deployment
- **Compromised Servers**
  Malicious servers manipulate outputs, intercept tool calls, or exfiltrate data
- **Missing Authorization and Access Controls**
  LLMs access tools they shouldn't or connect to unverified sources
- **Credential Leaks and Data Exfiltration**
  Sensitive data accessed via tools is exposed in outputs or logs
- **Lack of Oversight and Logging**
  No record of agent tool use, making it hard to detect issues or misuse

Risks of coding assistants

Vulnerabilities have been discovered in coding assistants themselves — for example, in early 2025, the Legit Security research team found a significant vulnerability in coding assistant GitLab Duo. The AI assistant integrated into GitLab and powered by Anthropic's Claude contained a remote prompt injection vulnerability that could allow attackers to steal source code from private projects, manipulate code suggestions shown to other users, and even exfiltrate confidential, undisclosed zero-day vulnerabilities — all through GitLab Duo Chat.

In addition, the code AI coding assistants produce and integrate into the application may also be susceptible to vulnerabilities. Specifically, vulnerabilities often come from insecure suggested code being copy/pasted without human validation. In fact, Cursor has a "Yolo" mode that allows AI to run commands without asking for approval.

Risks of LLM agents for coding workflows

LLM agents introduce unique security risks, especially due to their autonomous and multi-step behavior. Specifically, chained autonomous behaviors can escalate risk — such as using external tools, downloading libraries, running commands, or self-modifying code.

In 2023, researchers reported Auto-GPT, an autonomous AI agent, could be manipulated through indirect prompt injection to execute arbitrary code. In one instance, when tasked with summarizing content from an attacker-controlled website, Auto-GPT was tricked into executing malicious commands.

INSIDE THE GITLAB DUO PROMPT INJECTION ATTACK

Attacker plants hidden instruction

In a comment, issue, merge request description, or code.

User asks GitLab Duo a question

A normal request like "review this merge request".

Duo uses the hidden prompt unknowingly

The AI assistant includes the attacker's instruction in its response.

Private source code leaked

AI response contains malicious HTML that renders automatically, leaking private code to the attacker.

Duo runs in the user's context, giving attackers access to what the user sees.

The Way Forward

There is change, there is risk, there are new requirements, but there's also opportunity. AI is changing what AppSec entails, while also creating an opportunity to make AppSec more effective and more efficient.

New AppSec requirements

AppSec in the age of AI requires new:

Discovery

AI visibility is now a key part of AppSec. The ability to identify AI-generated code, and where and how AI is in use in your software development environment has become critical.

Threat modeling

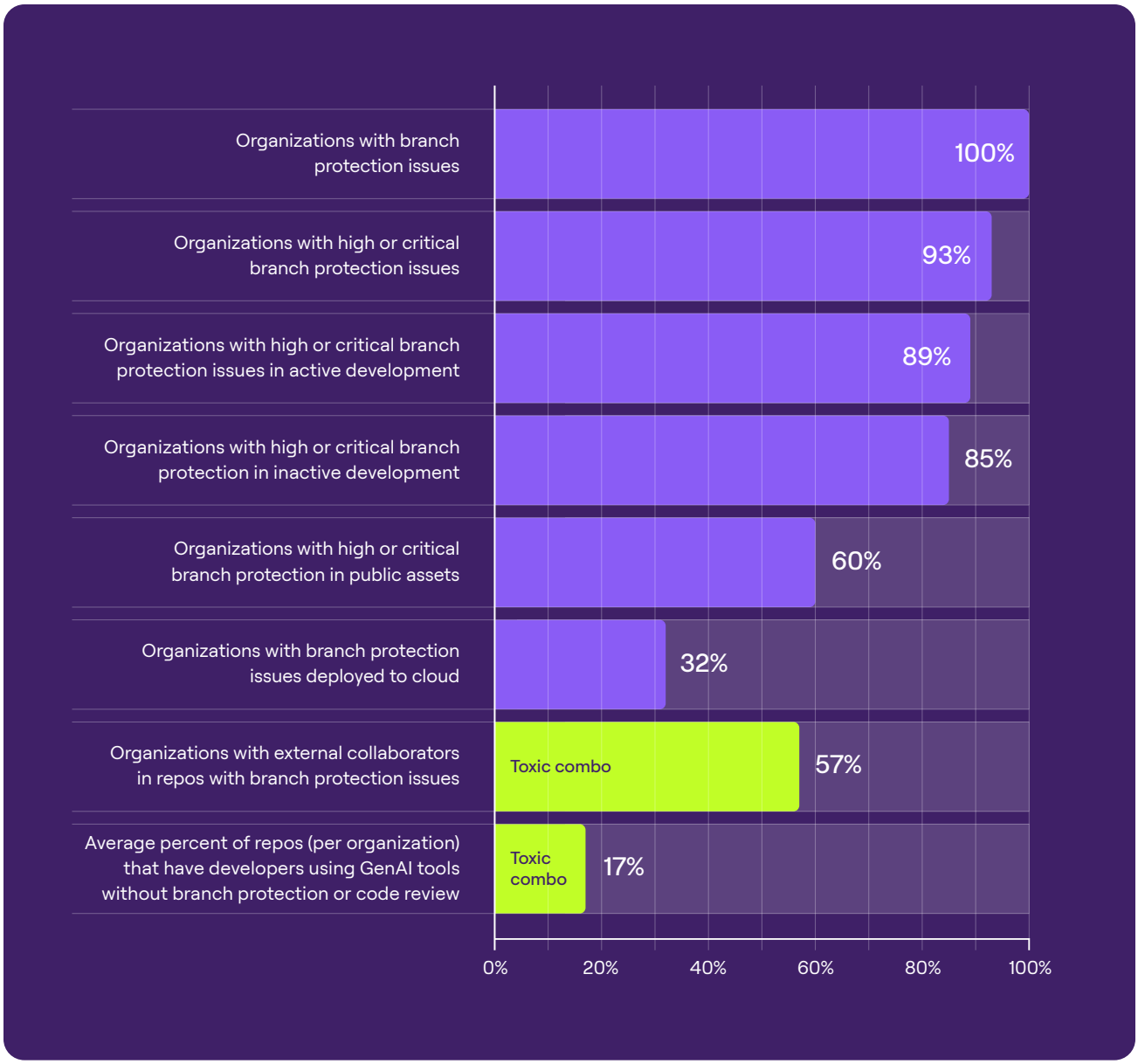
As the risk to the organization is changing, so too must threat models. If your app now exposes AI interfaces, is running an agent, or gets input from users and uses the model to process it, you've got new risks.

Security testing

AI-specific security testing has become vital. As mentioned above, AI brings in some novel vulnerabilities and weaknesses that traditional scanners can't find, such as training model poisoning, excessive agency, or others detailed in OWASP's LLM & Gen AI Top 10.

Awareness of toxic combinations

The use of AI in code development itself is not necessarily a risk. But when its use is combined with another risk, like lack of static analysis or branch protection, the risk level rises.



For instance, research for our 2025 State of Application Risk report revealed that, on average, 17% of repositories within organizations have developers using AI tools without proper branch protection or code review processes in place.

These "toxic combinations" require both discovering which development pipelines are using GenAI to create code, and then ensuring those pipelines have all the appropriate security measures and guardrails in place.

AI Creates New Security Opportunities

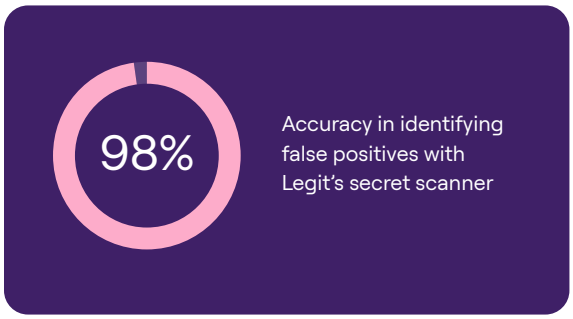
In the midst of new risks, AI brings real opportunity to transform and improve application security. Its ability to analyze massive amounts of data means teams can identify and prioritize vulnerabilities faster than ever. And if developers are already using AI assistants to write code, we have the chance to use those same assistants to catch issues early, embedding security into the development process itself. Finally, AI can streamline some of the operational security tasks that bog teams down.

Find and prioritize risk faster

AI excels at tasks that involve reviewing and analyzing a lot of data and surfacing connections and insights. It therefore shines in situations where there is a lot of data about risk and the need for quick insights or removal of false positives.

For example, secrets scanning is an ideal AI use case, since it's a language analysis problem, and one historically plagued by false positives. At Legit, we are leveraging AI to understand the nuance of secrets and when they should be considered false-positive. Given a secret and the context in which it was introduced, this model can establish the likelihood of a finding being a real threat or a false positive.

Using this approach dramatically reduces the number of false positives while keeping true positive rates stable. In fact, our secrets scanner is now approaching 98% accuracy. Finding risk and analyzing it will be a powerful task for AI in application security moving forward.



Shift even further left

With AI injected into the creation of code, there is also a new opportunity to embed automated security review and testing into early phases of code development. AI could make true shift-left security a reality. With security inserted into coding assistants, AI could conduct security review and remediation as it's generating code.

There are security issues that still need a human to address and analyze, but there are also classes of vulnerabilities that would be relatively simple and straightforward to bypass or circumvent with AI. Weaknesses like SQL injection will hopefully be a thing of the past as GenAI tools start to build secure-by-design rules into their models; however, these issues will be replaced with more complex issues that can now be identified by AI in business logic flaws, like IDOR, BOLA or auth/session type issues that have been very difficult to detect with automated security tools up until now.

Streamline AppSec operations

AI could relieve the AppSec team of some operational tasks that are currently critical, but also time-consuming and not adding value or reducing risk. For example, finding software owners. Because development teams are complex and dynamic, identifying owners is a time-consuming, manual task. But when a major vulnerability is announced, knowing who should own its remediation is critical information that needs to be readily accessible. Imagine the time saved if AI could automate this task and take over the tedious job of investigating and documenting code and app ownership.

How Legit Is Enhancing ASPM With AI

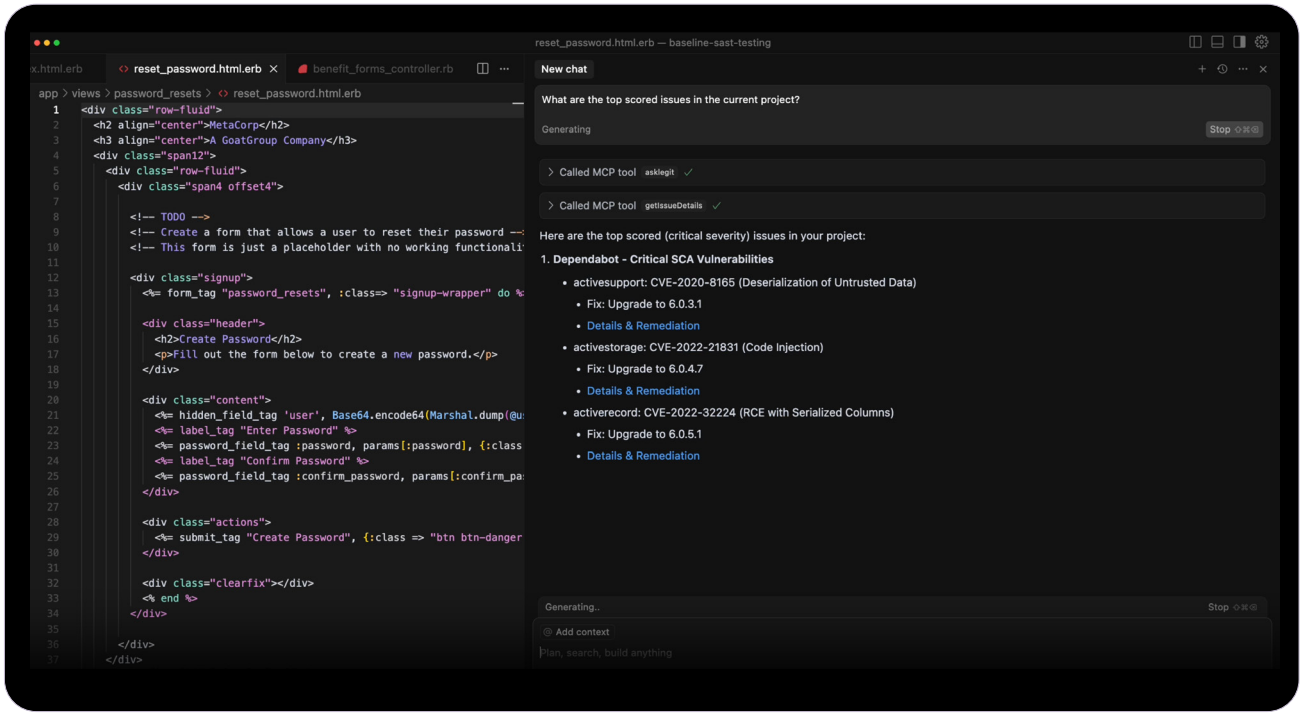
At Legit, we are infusing AI into everything we do — including what we secure and how we secure it. We are aiming to sit alongside your coding assistants and keep code secure while you write it, help you understand where you have AI writing code and keep it secure, and automate your AppSec with AI agents.

Our primary AI initiatives include:

Be your AppSec copilot

You code with an assistant, we'll add the security. Through the Legit MCP Server, AI code assistants like Cursor, GitHub Copilot, and Windsurf leverage Legit to determine the security of generated code, enforce guardrails to prevent issues, and drive automated remediation.

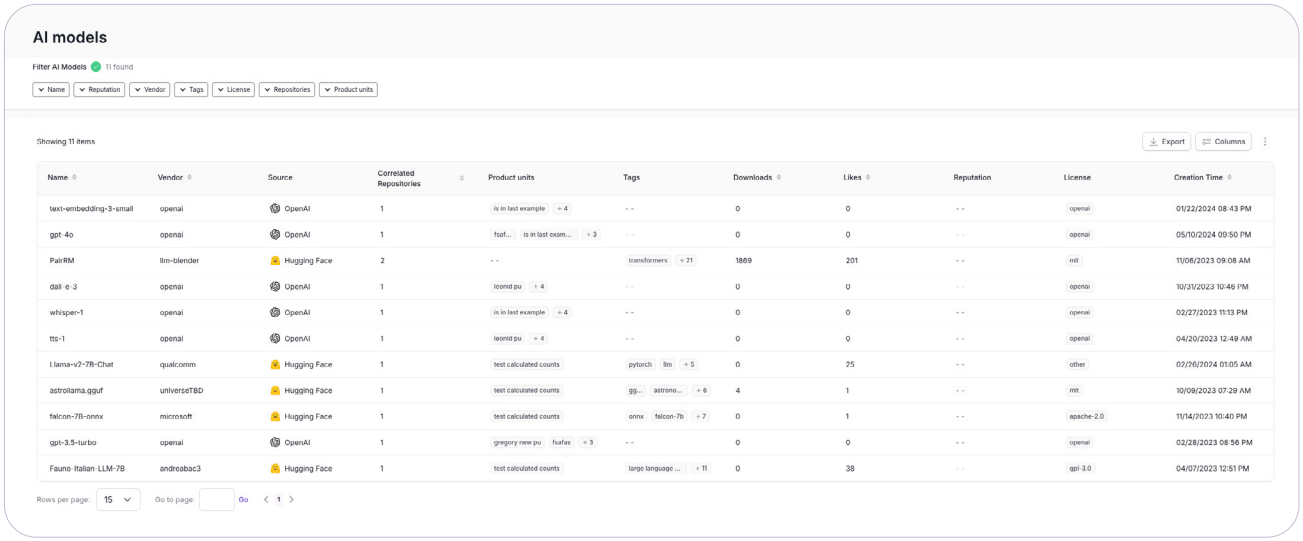
Developers get real-time security insights and remediation embedded directly into their AI-powered assistants — no learning curve, no context switching, all in natural language.



Legit MCP Server in Action

Secure the AI apps you build

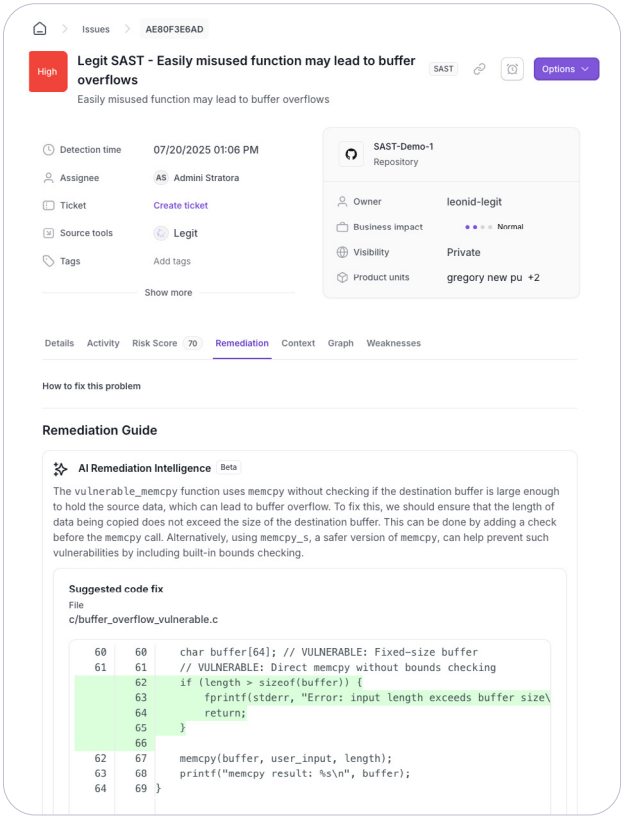
Legit provides full discovery and monitoring of all AI — off-the-shelf models and in-house MCPs. In addition, because traditional SAST tools don't catch AI risks like prompt injection, insecure model use, or missing validation, we offer AI SAST, which detects AI-specific code security issues.



Legit AI Discovery

Act as your AppSec AI agent

We're automating AppSec with AI to make it more effective and adapt to the AI tech stack. Legit prioritizes security findings, then provides contextual, AI-based remediation code snippets to simplify and speed up remediation, as well as the ability to open PRs directly via Legit. Additionally, you can query the Legit platform with natural language, making it simple to search, create reports, and take actions.



Legit AI Remediation

Future-Proof Your AppSec

Traditional AppSec tools and techniques are quickly becoming obsolete. Move forward with AppSec that can identify and secure AI-generated code and keep up with the current speed and volume of software development.

Related Resources

Learn more about how Legit is both leveraging AI and securing its use.

Request a Demo

Contact us for a demo of our powerful AI capabilities.

LEGIT

The Legit Security ASPM platform is a new way to manage application security in a world of AI-first development, providing a cleaner way to manage and scale AppSec and address risks. Fast to implement, easy to use, and AI-native, Legit has an unmatched ability to discover and visualize the entire software factory attack surface, including a prioritized view of AppSec data from siloed scanning tools. As a result, organizations have the visibility, context, and automation they need to quickly find, fix, and prevent the application risk that matters most. Spend less time chasing low-risk findings, more time innovating.

legitsecurity.com