LEGIT

# The AppSec Time-to-Exploit Problem

Why Application Security Is Losing an AI War Most CISOs Don't Know Has Started

# Executive Summary

## Something in application security has snapped, and most programs are still operating like nothing changed.

For roughly twenty years, AppSec ran on a simple premise: when a vulnerability is found, there is time to fix the issue before an attacker exploits it. That assumption is the foundation of every scanning tool, triage queue, SLA, and board slide that reads “we remediate critical findings within 30 days.”

That premise no longer holds. Google’s Threat Intelligence Group estimates that the average time-to-exploit across tracked vulnerabilities went negative in 2024, then fell to an estimated -7 days in 2025.[1]

In plain terms, attackers are often turning disclosures into weapons before a patch is public. At the same time, the average enterprise still needs about 74 days to remediate a critical or high-severity vulnerability. The gap between those two numbers won’t close by better execution. It’s a fundamental collapse of the AppSec model we all know.

The input to security teams has changed too, both in quantity and in character. AI is now responsible for an estimated 30% of code at Microsoft and as much as 75% of new code at Google.[3] Veracode research found AI-generated code contains 2.74x more vulnerabilities than human-written code, and 45% of AI-generated samples introduced a known OWASP Top 10 flaw.[4] So you get more code, produced faster, with a higher defect rate, reviewed by essentially the same-sized teams, while attackers no longer need a patch window to move.

### Legacy AppSec math doesn’t work for AI

- Code Volume ↑
- Code Generation Speed ↑
- Defect Rate ↑
- Speed of Attacks ↑
- Security Review Team →

## AI-generated code contains 2.74x more vulnerabilities than human-written code

This paper is about the math problem every CISO owns now, whether it’s on the board deck or not: an AppSec operating model built for a world where defenders had time doesn’t function in a world where they don’t. We’ll cover how the gap opened, what it’s already costing organizations that haven’t closed it, and what some security leaders are doing differently.

Part 01

# Three Trends That Converged Into One Crisis

No single event broke application security. Several trends, each significant on its own, ran into each other.

## 3 Trends Driving the AppSec Crisis

01 Code velocity and volume increased exponentially, while review capacity remains constant

02 AI-generated code is much more likely to carry risk than human-generated code

03 Code velocity and volume increased exponentially, while review capacity remains constant

Crisis 1

## Code velocity and volume increased exponentially, while review capacity remains constant.

AI coding assistants moved from novelty to core plumbing in under three years. Stack Overflow's 2025 developer survey reports 84% of developers now use AI tools regularly, and GitHub says AI-assistance accounts for 46% of code written on its platform, with projections of 60% by the end of 2026.[5] Microsoft's CEO publicly stated that AI generates 30% of the code in Microsoft's own repositories; Google's CEO says Google's code went from roughly 25% in 2024 to as much as 75% of new code by April 2026.[6]

None of that growth came with a matching rise in AppSec staffing. The same triage queues, the same manual steps, and often the same team size are now expected to absorb multiples of the code volume they were originally set up for.

Crisis 2

## AI-generated code carries more risk per line than human-written code.

This is the piece many boards still haven't fully internalized. AI driving an increase in code volume is only half the story. It also means a higher defect rate per line. Consider:

- Veracode's GenAI Code Security Report tested more than 100 large language models across four languages and found AI-generated code contains 2.74x more vulnerabilities than human-written code, with a 45% failure rate against OWASP Top 10 categories. It hit 86% for cross-site scripting specifically.[4]
- A separate December 2025 CodeRabbit analysis of 470 production pull requests found AI-authored code introduced vulnerabilities at almost double the rate of human-only commits, including a 57% higher rate of hardcoded credentials and exposed API keys.[7]
- Academic research from Stanford, Georgia Tech, and independent arXiv studies looking at hundreds of thousands of real-world commits point in the same direction, even where the exact multiplier varies by methodology.[8]

The practical takeaway is straightforward: the code entering pipelines now is produced faster, in larger volumes, and with a meaningfully higher likelihood of carrying an exploitable defect than the code most AppSec programs were designed around.

Crisis 3

## Attackers compressed their side of the equation to almost nothing.

Most teams still think in terms of a disclosure-to-exploit window. That window is shrinking to zero, and on average it has already flipped. As evidence:

- Mandiant's M-Trends 2026 report, based on more than 500,000 hours of incident response work across 2025, puts mean time-to-exploit at an estimated -7 days.[1]
- Exploitation of a known or zero-day vulnerability has also been the most common initial infection vector for six straight years, accounting for 32% of intrusions where Mandiant could attribute root cause.[9]
- Verizon's 2025 DBIR similarly reports vulnerability exploitation grew 34% year over year as an initial breach vector.[10]
- University of Illinois researchers showed GPT-4 could autonomously produce working exploits for 87% of one-day vulnerabilities using only the public CVE description, at roughly $8.80 per exploit.[12]

Attackers don't need long R&D cycles to weaponize disclosures. A description and a few hours (at most) can be enough.

The bottom line is we now see three trends with one outcome: more vulnerable code, arriving faster, with less time to act on it than at any point in the history of the discipline.

Part 02

# The Math No One Wants to Say Out Loud

Put attacker speed next to defender speed and it becomes hard to pretend this is a minor lag.

| | Attacker Side | Defender Side |
|---|---|---|
| **Time to act** | ~-7 days (exploits often precede public patches)[1] | 74 days average for critical/high findings[2] |
| **Volume trend** | 131 new CVEs disclosed per day[13] | Same or shrinking AppSec headcount relative to code volume |
| **Skill required** | Falling — AI can autonomously exploit disclosed CVEs[12] | Rising — every finding needs human triage, context, and a validated fix |
| **Cost of failure** | Near zero (an unsuccessful attempt costs the attacker little) | $4.44M average breach cost globally; $10.22M in the U.S.[14] |

This isn't "we're a little behind." A team running a 74-day remediation SLA against adversaries moving in single-digit days, and sometimes before patches exist, isn't competing in the same time frame. On vulnerabilities that matter, the defender is lapped before triage even starts.

## CVE disclosures are running at roughly 131 per day, up nearly 50% year-over-year

Mandiant explicitly notes that newly discovered vulnerability volume keeps outpacing remediation capacity, creating backlogs in the hundreds of thousands at large enterprises, and occasionally in the millions.[9]

Each day the backlog grows without fix throughput rising to match, the odds increase that the next major incident comes from something already in the queue, not something newly found. Most breaches attributed to vulnerability exploitation aren't unknowable zero-days. They're known issues that were still sitting in backlog when an attacker first arrived.

Part 03

# Why “Shift Left” and Find-It-Fix-It Can’t Close This Gap Alone

None of this is an argument against detection. Detection has improved dramatically. SAST reads commits, SCA tracks dependencies, secret scanners and IaC checkers run continuously, and reachability work covers more ground than ever.

The failure isn’t detection. It’s in what happens next.

The old workflow — find it, ticket it, hand it to a developer, wait for a fix, validate, merge — assumed findings arrived slowly and defenders had weeks or months. AI-era volume and attacker speed expose three structural weaknesses:

01 **It assumes every finding is worth human attention.**

Legacy tools generate huge piles of findings without the application-level context (e.g., business criticality, data sensitivity, exploitability, production status) needed to know which ones are urgent. Without those signals, everything looks urgent, which in practice means nothing is.

02 **It forces a human to reconstruct context a scanner discarded**

Every finding a developer picks up requires them to rebuild an understanding of what the code does, why it’s shaped that way, what depends on it, and what breaks if it changes. This is all work the scanner could have preserved but didn’t. This is why remediation, not detection, has become the actual bottleneck: the manual chain from finding to merged fix hasn’t materially changed in over a decade, even as everything upstream of it has accelerated by an order of magnitude.

03 **It has no mechanism to make the next cycle better**

A ticket closed today teaches the system nothing about the next similar finding. Each fix is an isolated transaction, not a data point that improves how code is generated or how future issues are triaged. There’s no cumulative improvement because the backlog resets every single day.

The result is arithmetic that only ever moves one direction: a linear, human-paced remediation process running against a parallel, AI-accelerated detection and attack surface. The backlog does not stabilize. It grows, and the highest-risk issues, which are also usually the hardest to fix, are exactly the ones that sit and age the longest.

Part 04

# What's Actually at Stake (Beyond the Security Team)

This isn't a problem that stays contained to the AppSec function, and that's precisely why it deserves board-level attention now rather than after an incident forces the conversation.

**Cost of a data breach in the U.S.: $10.22 million**

### The financial exposure is already quantified

IBM's 2025 Cost of a Data Breach Report puts the global average cost of a breach at $4.44 million and $10.22 million in the United States, an all-time high.[14] Breaches that take longer than 200 days to identify and contain cost, on average, $1.88 million more than those caught faster.[15] Given that the industry-wide median time to remediate a known vulnerability already runs into the hundreds of days, a meaningful share of that exposure is self-inflicted by remediation speed alone, rather than detection failure.

### The regulatory and personal liability exposure is new, and it's specific to leadership.

Since 2023, SEC rules require public companies to disclose material cybersecurity incidents within four business days of determining materiality, and to describe how the board governs cyber risk in annual filings.[16] Legal and governance experts increasingly describe board cybersecurity oversight through the lens of Caremark fiduciary duty. This means directors and officers can face personal liability for failing to implement any reasonable system for monitoring cyber risk, not just for the incident itself.[17] Several CISOs have already been personally named in enforcement actions tied to inadequate breach disclosure or unverifiable compliance claims.[18] "We had a scanner running" is a materially weaker defense than it was three years ago. Regulators and plaintiffs' counsel are now asking what happened after the scanner found something.

### The compounding effect is the part most programs underweight

A vulnerability that sits in a backlog for 74-252 days is a growing risk, not a static one. Every day it survives, the probability that an increasingly AI-augmented attacker population finds and exploits it rises, while the cost of the eventual breach doesn't fall. Leadership teams that treat their remediation backlog as a static operational metric are underpricing the single biggest driver of breach cost they can control.

Part 05

# The Shape of the Answer – Agentic AppSec

Gartner's Hype Cycle for Secure Software Engineering formally recognized two adjacent, converging categories in 2026: Agentic Coding Security and Application Security Posture Management. This reflects an industry-wide acknowledgment that context, remediation, and prevention can no longer be solved as separate problems by separate tools.[19] The common thread across the leading thinking in this space is a shift away from "find it, ticket it, hope someone gets to it" toward a continuous, machine-speed loop with three properties:

### Context, not just findings

A vulnerability isn't a fixed quantity of risk. Its urgency depends entirely on where it exists: is the affected application Internet-facing? Does it handle sensitive data? Is the vulnerable code path reachable in production? Without that context, every finding looks equally urgent, which in practice means none of them do. With it, the handful of findings that are genuinely dangerous can be separated from the noise instantly, instead of after a human spends an hour investigating.

### Remediation at the speed detection already runs

If scanners can generate findings continuously and autonomously, the fix side of the equation has to run the same way. Agents read the surrounding code, understand the vulnerability class, generate a fix that addresses the root cause rather than the symptom, and validate that the fix holds before it ever reaches a human for approval. The goal isn't removing developers from the loop; it's removing the hours of manual context-reconstruction that currently stand between a finding and a fix.

### A feedback loop back into how code gets written

The most durable version of this model doesn't stop at remediation. Every fix an agent makes is a data point about a mistake that keeps recurring. That insight can be fed back into the AI coding tools generating new code in the first place, so the same class of vulnerability shows up less often with each cycle. This is what separates a point solution from a compounding one: a program that gets measurably better over time instead of resetting to zero every sprint.

---

This is the model an emerging set of practitioners are calling agentic AppSec. Rather than AI bolted onto an existing scanner, agentic AppSec reflects a genuinely different operating model built for a world where both code and attacks move at machine speed. Legit Security has been building toward this model since before the term existed: a platform architected around the belief that context, remediation, and prevention have to work as one continuous system, not three disconnected tools stitched together with tickets.[20]

Part 06

# What to Ask Your Team This Quarter

As you settle into this new reality, a few questions worth putting in front of your AppSec team this quarter:

- What is our actual median time from finding to merged fix? Not our SLA, our real number for critical and high findings, and how does that compare to the -7-day to 5-day time-to-exploit window the industry is now operating in?
- How many findings in our current backlog have been open longer than 90 days, and do we know which of those sit in Internet-facing, data-sensitive, or otherwise high-blast-radius applications?
- If a critical vulnerability is disclosed in a library we use tomorrow, how long would it take us to know which of our applications are affected (not just whether we use the library, but whether the vulnerable code path is reachable)?
- Does our remediation process teach our AI coding tools anything, or does every fix start from zero context the next time a similar issue appears?
- Could we produce, within four business days, a defensible account of our vulnerability management posture if a board member or regulator asked for one tomorrow?

If any of those questions produce a long pause instead of a number, that pause is the finding.

# About Legit Security

Legit Security is the Agentic Application Security company, purpose-built for a world where AI writes code. Legit's platform autonomously prioritizes and remediates the vulnerabilities that matter most and prevents new ones from being introduced at the moment AI code is generated. Legit continuously learns from your codebase to deliver secure AI-generated code at a speed and scale no traditional AppSec approach can match. Legit is trusted by security teams worldwide, including the Fortune 500, and is rated 4.8 on Gartner Peer Insights.

legitsecurity.com

## Sources


1. Mandiant, M-Trends 2026: Data, Insights, and Strategies From the Frontlines, Google Cloud, April 2026.
2. Industry vulnerability remediation benchmarks; average critical/high remediation time and industry-wide median remediation time, as reported across 2025-2026 application security research.
3. Statements by Microsoft CEO Satya Nadella (LlamaCon, April 2025) and Google CEO Sundar Pichai (public earnings commentary, 2024-2026) on the share of AI-generated code in company repositories.
4. Veracode, 2025 GenAI Code Security Report.
5. Stack Overflow, 2025 Developer Survey; GitHub, 2025 research on Copilot-assisted code share.
6. See note 3.
7. CodeRabbit, State of AI vs. Human Code Generation, December 2025 analysis of 470 production pull requests.
8. Academic research including Stanford University (Perry et al.), Georgia Tech SSLab's "Vibe Security Radar," and large-scale arXiv empirical studies of AI-generated code in public repositories, 2022-2026.
9. Mandiant M-Trends 2026; initial infection vector data.
10. Verizon, 2025 Data Breach Investigations Report.
11. Industry vulnerability statistics aggregating Mandiant and third-party application security research, 2026.
12. University of Illinois Urbana-Champaign research on LLM-autonomous exploitation of disclosed CVEs, as widely reported in 2024-2026 security research.
13. CVE disclosure volume tracking, 2025-2026 industry reporting.
14. IBM, Cost of a Data Breach Report 2025, in partnership with Ponemon Institute.
15. IBM, Cost of a Data Breach Report 2025 — breach lifecycle cost analysis.
16. U.S. Securities and Exchange Commission, cybersecurity disclosure rules (Release No. 33-11216), effective 2023, Item 1.05 of Form 8-K.
17. Legal and governance analysis applying Caremark fiduciary duty standards to cybersecurity oversight, as discussed in 2025-2026 corporate governance publications.
18. SEC enforcement actions involving individual CISO liability for breach disclosure and internal controls, 2024-2026.
19. Gartner, Hype Cycle for Secure Software Engineering, 2026 — Agentic Coding Security and Application Security Posture Management category definitions.

Legit Security is the AppSec platform purpose-built to secure AI-powered development. Our AI-native ASPM secures modern software development, including AI-first pipelines, code assistants, agents, and vibe coding. With unmatched visibility across the SDLC and from code to cloud, Legit makes it easy to identify, prioritize, and fix AppSec issues that matter most to the business.

LEGIT