Vibe-Coded and Vulnerable:

A Benchmark Study of Agentic Security Tooling on AI-Generated Applications

Zauth, Inc. | May 2026 | zauth.inc

Abstract

The proliferation of AI-generated web applications has outpaced the security tooling designed to protect them. We present a controlled benchmark study comparing two agentic penetration testing platforms across ten production-grade applications built on Lovable, a leading AI-powered development platform. Both tools were deployed against the same targets on the same day: Vector, a black-box agentic scanner with usage-based pricing averaging $14.31 per scan, and Aikido Security, a white-box pentesting platform backed at a billion-dollar valuation and priced at $100 per assessment. Across 141 normalized root-cause vulnerabilities identified, Vector detected 94% of critical findings operating without source code access. Aikido, with full source code access, detected 47% of critical findings. The cost per weighted severity point was $0.36 for Vector versus $3.73 for Aikido, a more than 10x efficiency gap. Vector completed assessments in an average of 34.7 minutes versus 69.7 minutes for Aikido. The security failure modes of AI-generated applications are systematic, detectable, and underserved by tooling at the price points and speeds developers can actually afford.

1. Introduction

Software development is undergoing a fundamental shift. AI-assisted development platforms like Lovable, Replit, Bolt, and Base44 have made it possible for non-engineers to build and deploy production web applications in hours. Collectively these platforms serve over 80 million users, generate $1.5 billion in annual recurring revenue, and see 100,000 new projects built every day. These are not toys. They are production platforms at scale, and the applications they generate handle real users, real data, and real money.

This shift has a security problem nobody is talking about.

AI-generated code inherits the patterns its training data emphasizes. It scaffolds authentication flows, database schemas, and API endpoints at speed, but it does not reason about threat models. The developer using these platforms is, by definition, not a security engineer. They are a founder, a designer, a student, someone with an idea and an internet connection who discovered they can build software without writing it. Security is not their first question. Often it is not their question at all.

The result is a new class of production application: feature-complete, user-facing, and largely unaudited. These applications share a common technology fingerprint. The attack surface they generate is predictable, structural, and broad enough to be dangerous at scale.

The industry has begun to notice. In March 2026, Lovable became the first AI development platform to integrate native security scanning into its product, partnering with Aikido Security, a leading agentic

pentesting platform backed at unicorn valuation, to offer automated assessments at $100 per scan. It was a meaningful step. The largest and fastest-growing AI-powered development platform in the world looked at the security problem and picked a solution.

We wanted to know if that solution was good enough.

To find out, we built ten production-grade applications using Lovable and subjected each to concurrent assessment by two agentic penetration testing platforms: Vector, a black-box scanner with usage-based pricing, and Aikido Security, operating at $100 per assessment. Both tools ran against identical targets on identical days. All findings were normalized by root cause, scored using CVSS v3.1, and deduplicated across tools.

What we found was consistent enough across all ten sites to constitute a pattern. AI-generated applications built on this stack have a predictable security fingerprint. The vulnerabilities are not random. They are structural, and in several cases directly exploitable against real users and real assets.

This paper documents that fingerprint, quantifies the detection gap between two classes of security tooling, and argues that the scale of AI-generated code deployment demands a rethinking of how security assessment happens and at what price.

2. Methodology

2.1 Target Selection

All ten target applications were built using Lovable between March and May 2026. Applications were selected to represent the range of use cases typical of AI-generated production software: a social networking platform, a cryptocurrency betting exchange, a real estate marketplace, an NFT trading platform, a project collaboration tool, an educational blockchain platform, a multi-tenant invoicing SaaS, a healthcare appointment booking system, a bounty and task management platform, and a CRM with AI-powered features. Each was deployed to a live production URL with real functionality, real authentication, and real data storage. No artificial vulnerabilities were introduced. The applications were assessed exactly as a real attacker would encounter them.

All ten applications share the default Lovable technology stack: React and Vite on the frontend, Supabase for database and authentication, and serverless edge functions for business logic and AI integrations.

2.2 Tools

Vector is a black-box agentic security scanner developed by Zauth, Inc. It operates without source code access, using browser automation, JavaScript analysis, endpoint discovery, and exploitation verification to identify vulnerabilities. Vector uses usage-based pricing that scales with the depth of the attack surface. Actual costs across the ten assessments ranged from $7.20 to $19.34 per scan, with a total spend of $143.12 and an average of $14.31. Vector was run in Deep Scan mode on each target. All findings were verified for reproducibility post-scan. The false positive rate in this study was zero.

Aikido Security is a white-box agentic pentesting platform integrated as the native security solution within Lovable. Aikido operates with full access to application source code and follows a structured

OWASP-aligned methodology. The $100 per scan pricing reflects Aikido's Lovable-specific integration; their standard platform pricing begins at $1,000 to $4,000 per engagement, making the real-world cost gap substantially larger than this study reflects. This structural advantage (full source code visibility) is discussed in detail in Section 5.

Both tools were run against each target on the same day. Neither was given any advantage in terms of credentials, prior knowledge, or configuration beyond what each platform provides by default.

Disclosure: This study was conducted by Zauth, Inc., the developer of Vector. Full raw reports from both tools are available to researchers and investors on request for independent verification. Contact: team@zauth.inc

2.3 Scoring and Normalization

All findings were scored using CVSS v3.1. Severity bands: Critical (9.0 to 10.0), High (7.0 to 8.9), Medium (4.0 to 6.9), Low (0.1 to 3.9). Informational findings were excluded from all counts and cost calculations.

Raw findings were normalized by root cause prior to analysis. Multiple findings sharing a single root cause and remediation path were counted as one finding. Findings identified by both tools against the same target were counted once and attributed to both. Findings identified by only one tool were counted as unique to that tool.

2.4 Cost and Time Inputs

The table below records the exact per-scan cost and duration for each of the ten assessments. Vector uses usage-based pricing; Aikido charges a flat $100 per assessment.

Site	Type	Vec. Cost	Aik. Cost	Vector Time	Aikido Time
nexusxhub.online	Social	$17.49	$100.00	22 min	25 min
pumpbet.site	Crypto betting	$7.20	$100.00	25 min	37 min
easyhouse.casa	Real estate	$10.96	$100.00	38 min	73 min
kaleidochain.site	NFT platform	$11.62	$100.00	41 min	73 min
forgesflow.online	Collaboration	$18.91	$100.00	33 min	64 min
learnchain.site	Ed-tech	$15.12	$100.00	40 min	82 min
invoicechain.site	Invoicing SaaS	$19.34	$100.00	50 min	77 min
medicarebook.site	Healthcare	$14.41	$100.00	34 min	64 min
bountyboard.site	Task mgmt.	$14.12	$100.00	34 min	116 min
trustcrm.site	CRM	$13.95	$100.00	30 min	86 min
Total / Average		$143.12	$1,000.00	34.7 min avg	69.7 min avg

Table 1. Actual per-scan costs and scan duration by site. Vector uses usage-based pricing; Aikido charges a flat $100 per assessment.

3. The Attack Surface: What AI-Generated Apps Look Like

Every application in this study was generated by Lovable and deployed to production without modification to the underlying architecture. The result in each case was the same: a React single-page application bundled with Vite, a Supabase project handling authentication, database, and file storage, and between two and five serverless edge functions for business logic and AI integrations. Cloudflare sat in front of each application as a CDN and proxy layer.

This stack is not unique to Lovable. Bolt, Replit, and Base44 generate nearly identical architectures by default. The attack surface documented in this paper is not a product of one platform's choices. It reflects how AI code generation currently handles security across the ecosystem. The findings documented here are consistent with hundreds of additional deployments on these platforms outside this study.

Both tools assessed between 17 and 25 endpoints per application, covering authentication flows, REST API layers, serverless edge functions, and client-side JavaScript bundles. The attack surface was identical for both tools on every target.

This study focuses on ten Lovable-built applications sharing a consistent Supabase-backed stack. Findings should be interpreted in that context rather than as a universal claim about all web application security tooling. A larger multi-platform comparative study is in progress.

4. Results

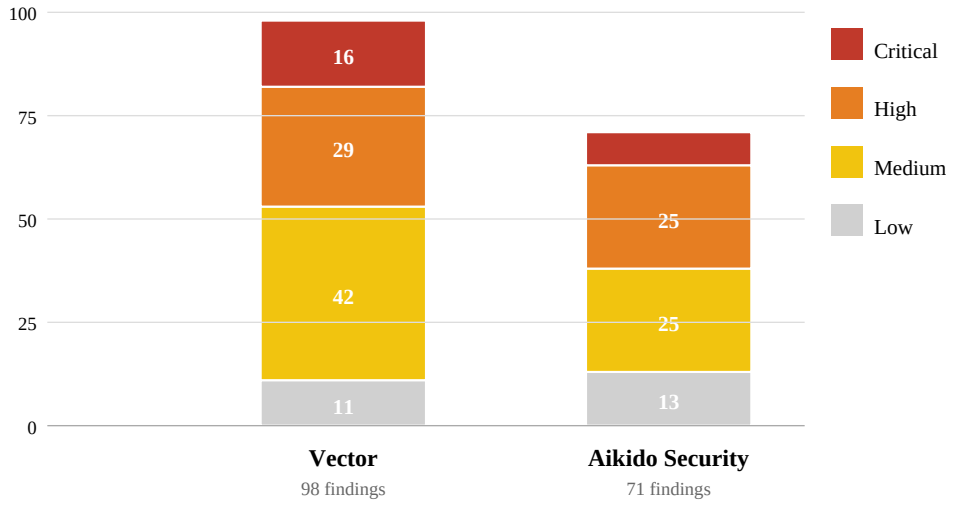
4.1 Overview

Across ten applications and twenty concurrent assessments, Vector identified 98 normalized root-cause vulnerabilities compared to 71 for Aikido. Vector found 38% more vulnerabilities at roughly 14% of the cost and in less than half the time. Of the 17 critical findings identified across all ten sites, Vector detected 16 (94%) and Aikido detected 8 (47%). Vector uniquely identified 9 critical vulnerabilities Aikido missed entirely, operating without access to a single line of source code. Aikido, with full source code access, uniquely identified 1 critical vulnerability Vector missed, a cross-tenant workspace takeover on invoicechain.site requiring direct inspection of RLS policy logic.

Metric	Vector	Aikido Security
Total normalized findings	98	71
Critical findings	16	8
High findings	29	25
Medium findings	42	25
Low findings	11	13
Unique findings	70	43
Confirmed by both tools	28	28
Severity-weighted score	400	268
Total cost	$143.12	$1,000.00
Cost per finding	$1.46	$14.08
Cost per weighted severity point	$0.36	$3.73
Total scan time	347 min	697 min
Average scan time	34.7 min	69.7 min

Table 2. Aggregate findings, cost, and time comparison across all ten sites.

Figure 1. Severity Distribution by Tool



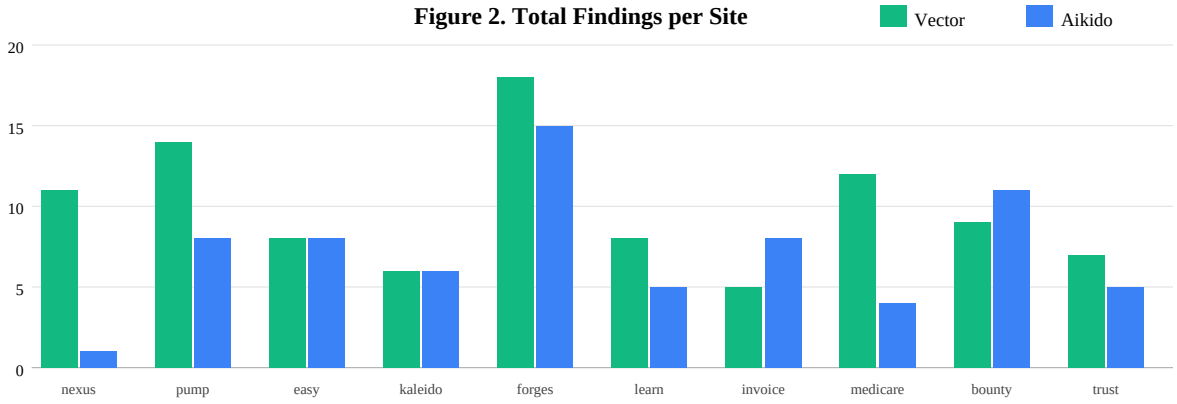
4.2 Site by Site

Results were consistent in their asymmetry across most sites, with Vector outperforming on raw finding counts, critical detection, and cost efficiency in 8 of 10 cases. Two sites merit specific framing.

On invoicechain.site, a multi-tenant invoicing SaaS, Aikido identified 8 findings to Vector's 5. Three separate cross-tenant data exposure vulnerabilities, including a workspace takeover via a mutable company identifier in the profiles table (CVSS 9.1), were accessible only through direct source code review. This result is included unmodified and addressed directly in Section 5.3.

On bountyboard.site, Aikido identified 11 findings to Vector's 9. Aikido's findings were predominantly authenticated authorization defects requiring source code understanding of intended task permissions. Despite having source access, Aikido missed the only XSS finding on the site, which Vector identified via JavaScript bundle analysis.

Figure 2. Total Findings per Site



Per-site finding details are documented in Appendix A. Scan durations for each assessment are listed in Table 1.

4.3 Critical Findings

The most consequential gap between the two tools is at the critical severity level. The following findings were identified by Vector and not detected by Aikido, which had full access to the source code of every application.

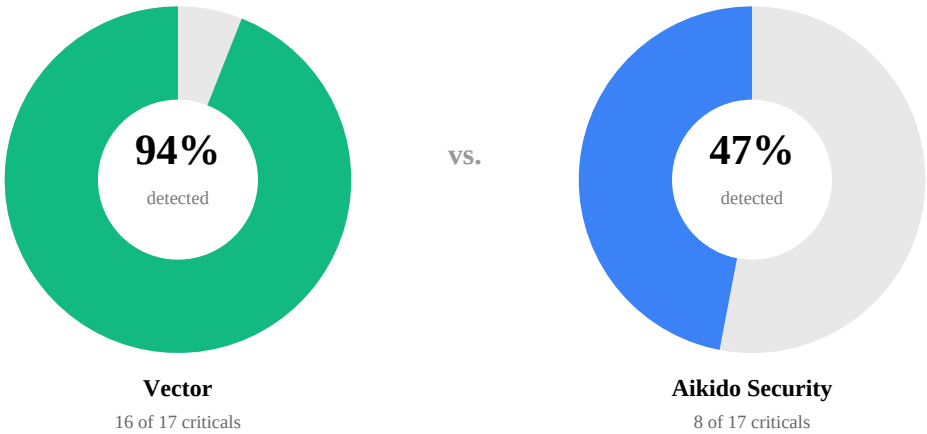
On nexusxhub.online, Vector identified a broken RLS policy exposing the personally identifiable information of over 30 users to any authenticated account, and a stored XSS vulnerability introduced by the application's AI chat feature rendering LLM output as raw HTML without sanitization.

On pumpbet.site, Vector identified encrypted Solana private keys stored in a Supabase table readable by any unauthenticated internet user, a direct path to wallet theft for every custodial user on the platform. Vector also identified a complete RLS bypass across all tables and a prompt injection vulnerability that extracted the application's full system prompt via role injection.

On kaleidochain.site, Vector identified an unauthenticated admin panel exposing full platform data; unauthenticated write access allowing any internet user to insert records into any table and impersonate any wallet; a complete unauthenticated data dump via the Supabase REST API; and a sybil attack and vote stuffing chain demonstrating financial manipulation of royalty distributions on a live NFT platform. This chain required linking four separate vulnerabilities and was not detected by Aikido despite its full visibility into the source code.

On medicarebook.site, Vector identified the only critical finding on the site: a stored XSS vulnerability via the MediBot AI chat widget rendering unsanitized HTML (CVSS 9.3), found through JavaScript bundle analysis. Aikido had full source code access to this application and did not identify the vulnerability. This is the second site in this study where Aikido missed an AI-rendered XSS that Vector found via black-box JS inspection, despite having the rendering logic directly available in source.

Figure 3. Critical Finding Detection Rate



4.4 What Aikido Found That Vector Missed

Intellectual honesty requires stating this clearly. Aikido identified 43 unique findings Vector did not catch, including 1 critical finding. These cluster in two categories: business logic vulnerabilities that require knowing what the application was designed to do, and infrastructure configuration issues.

Notable unique findings from Aikido include a race condition in a cryptocurrency wallet function allowing unlimited credit minting via concurrent API calls, a non-atomic marketplace purchase flow allowing listings to be forced to sold status without completing an ownership transfer, three separate cross-tenant data exposure vulnerabilities on invoicechain.site accessible only through source code inspection, and authenticated authorization defects on bountyboard.site requiring direct knowledge of intended task permission boundaries.

These are real vulnerabilities. They represent a class of finding that requires source code access to identify reliably. With equivalent source visibility, Vector would be expected to surface these alongside everything it already finds dynamically.

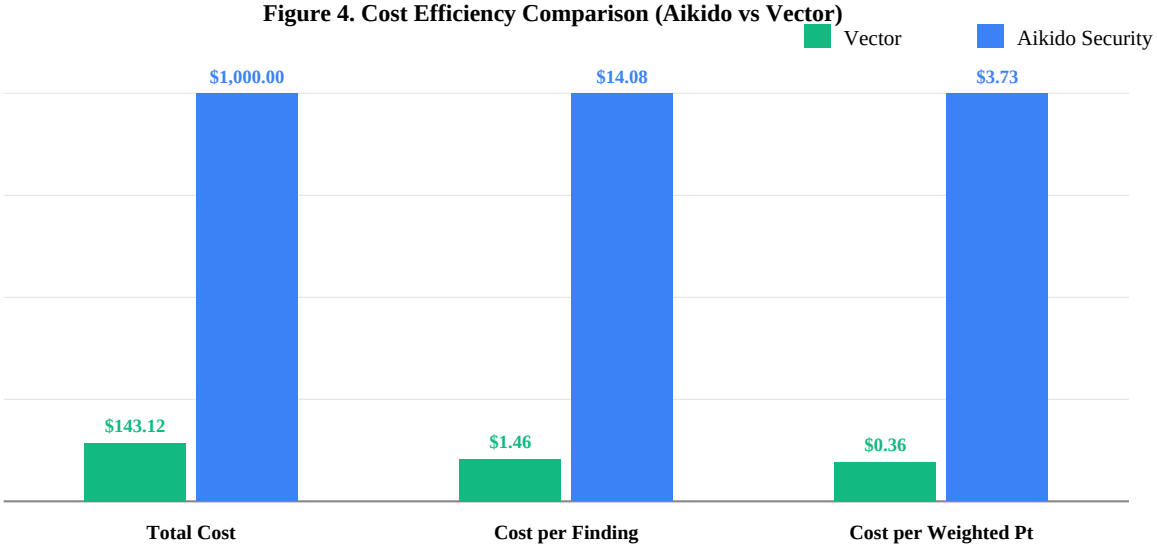
4.5 Cost, Speed, and Efficiency

The performance gap extends across three dimensions: detection rate, cost, and speed. Vector delivered findings at $1.46 per normalized vulnerability compared to $14.08 for Aikido, nearly a 10x gap per finding. When weighted by severity, Vector delivered $0.36 of detection per dollar spent compared to $3.73 for Aikido, a more than 10x efficiency gap. Operating without source code access, at roughly 14% of the total cost, Vector identified 94% of critical findings to Aikido's 47%.

Vector was also significantly faster. Total scan time across all ten sites was 347 minutes for Vector versus 697 minutes for Aikido. On several sites (bountyboard.site, trustcrm.site, and learnchain.site), Vector completed its assessment in less than half the time Aikido required. Speed matters because AI-generated applications ship in hours, not days. A tool requiring up to 116 minutes per deployment cannot integrate into a same-day shipping workflow.

For a developer shipping on Lovable, the practical choice is between a sub-$20 scan averaging 35 minutes that catches 94% of critical vulnerabilities, and a $100 scan averaging 69.7 minutes that catches 47%, or no scan at all. Across the 100,000 applications built daily on platforms like Lovable, Replit, and Bolt, that difference determines whether security assessment happens at all.

Figure 4. Cost Efficiency Comparison (Aikido vs Vector)



5. Finding Quality Analysis

5.1 Exploitation Depth

Security tooling is typically evaluated on finding counts and false positive rates. This paper argues a third dimension matters more: finding consequence. Not just whether a vulnerability was identified, but what an attacker could actually do with it, and whether the evidence makes that consequence undeniable.

Both tools produced real findings with real impact. Aikido's race condition on pumpbet.site and its cross-tenant takeover findings on invoicechain.site are not theoretical. They represent genuine paths to financial loss and data exposure, documented with clear reproduction steps and accurate severity scores.

Where Vector's reports differed was in active exploitation evidence. On nexusxhub.online, Vector demonstrated rendered JavaScript execution via LLM output, providing working exploit code rather than a potential vector. On pumpbet.site, Vector returned the actual encrypted private keys of real custodial wallets as evidence. On kaleidochain.site, Vector walked the complete sybil attack chain: account creation, signature bypass, vote inflation, and royalty manipulation, demonstrating the full path from initial access to financial impact.

This matters for two reasons. First, it eliminates false positives by definition: a finding backed by working exploit evidence is not a false positive. Second, it communicates risk in terms a non-security developer can act on immediately. A developer who reads "potential RLS misconfiguration, CVSS 8.8" may not grasp the severity. A developer who sees their own private keys returned as evidence does.

5.2 Finding Categories

Examining where each tool found unique vulnerabilities reveals a clear pattern. Vector's unique findings concentrate in three categories: database access control failures exploited to their logical conclusion, cross-layer attack chains requiring linking of multiple weaknesses, and AI-specific vulnerabilities including prompt injection, LLM output rendering flaws, and unauthenticated AI endpoint abuse. Aikido's unique findings concentrate in two categories: business logic flaws requiring knowledge of intended application behavior, and infrastructure configuration issues such as TLS cipher suites, cookie flags, and authentication workflow bypasses. Full root cause distribution by class is detailed in Appendix C.

5.3 The White-Box Advantage and What It Did Not Prevent

White-box testing should, by definition, be strictly superior to black-box testing. A tool with full source code access can do everything a dynamic scanner does and then read the business logic directly on top of it. Aikido, backed by over $84 million in funding and operating with complete source code visibility on every target in this study, should have found everything Vector found and more. That is not what happened.

The invoicechain.site result shows where source code access genuinely matters. Three separate cross-tenant data exposure vulnerabilities, including a workspace takeover via a mutable company identifier in the profiles table, required reading RLS policy logic to understand that user-mutable fields were being used as tenancy boundaries. Vector did not surface these. Aikido did. Those results are included unchanged in this paper.

But across all ten applications, Aikido missed 9 of the 17 critical findings in this dataset. It missed private keys readable without authentication. It missed a complete sybil and vote manipulation chain on a live NFT platform. It missed an AI-rendered XSS in a healthcare application despite having the edge function source code and frontend rendering logic directly available. These are exposed endpoints, disabled access controls, and missing authentication gates, the kind of findings an attacker identifies in minutes. Vector found them without a single line of source code.

The implications run in both directions. For Aikido, the results suggest its dynamic coverage has meaningful gaps that source code access alone does not bridge. For Vector, source code access (whether through platform-native integration or manual export) would add the business logic layer Aikido has an edge on, while retaining the dynamic coverage Aikido demonstrably lacks.

5.4 The AI-Specific Finding Category

One finding category deserves separate attention because it did not exist as a meaningful attack surface two years ago. Every application in this study included AI-powered features, and every application exposed at least one unauthenticated endpoint through which those features could be abused.

Vector identified and demonstrated these vulnerabilities across all ten sites, including active proof of prompt injection, system prompt extraction via role manipulation, and financial denial of service through unauthenticated API credit consumption. The AI-specific finding count for Vector across the full dataset is 12, compared to 8 for Aikido. Aikido identified unauthenticated AI endpoints on several sites but did not demonstrate prompt injection or system prompt extraction on any site, and did not

quantify the financial exposure from unauthenticated credit consumption.

A pattern emerged across multiple sites that is particularly notable: Aikido consistently missed XSS vulnerabilities introduced through AI feature rendering, despite having source code access. Vector identified these on two separate sites through JavaScript bundle analysis of how AI output was being rendered to the DOM. Aikido had the source code for both edge functions and the frontend rendering logic and did not surface either finding. A thorough white-box review should have caught these. That it did not points to a gap in dynamic analysis coverage, not a limitation of the white-box approach itself.

6. The AI-Assisted Development Security Crisis

The findings in this paper do not represent a collection of individual developer mistakes. They represent a systemic failure of security at the point of production.

One hundred thousand new applications are built on AI-powered development platforms every day. The developers building them are, by design, not security engineers. They are moving from idea to deployed product in hours, often without a security review, a penetration test, or even a basic understanding of the attack surface their platform generates by default. The applications they ship are live, indexed, and actively serving real users before anyone has asked whether the database is locked down.

6.1 The Scale Problem

Security tooling has historically been built for security teams. Enterprise scanners, manual penetration tests, bug bounty programs: all of these assume an organization with security staff, security budgets, and defined security processes. The developer using Lovable or Replit has none of these things. They have a subscription, a domain name, and users who trust them with their data.

The economics of existing security tooling do not work at this scale. A manual penetration test costs thousands of dollars and takes weeks. Enterprise scanners require configuration, maintenance, and interpretation by people who understand the output. Even Aikido's $100 automated assessment, the tool Lovable chose as its native security solution, is priced above the point where a solo developer will pay for it on every deploy. And as this paper demonstrates, at $100 it still missed 53% of the critical vulnerabilities in this dataset.

Vector's usage-based pricing means a typical assessment costs less than $20 and scales with the complexity of the application being tested. At that price point, security assessment becomes economically viable at the scale of the problem.

6.2 What Needs to Happen

Lovable's integration of Aikido as a native security scanning solution is the right instinct. The platforms driving this wave are beginning to take security seriously, and that matters. But integrating a tool that missed 53% of critical vulnerabilities in this dataset does not fully solve the problem.

Closing this gap will not come from better developer education, stricter platform warnings, or more expensive enterprise tooling. It will come from security infrastructure that operates at the speed, price point, and accuracy of the platforms generating it.

7. Conclusion

This study set out to answer a simple question: is the security tooling Lovable chose to protect its users good enough for the problem it is trying to solve?

The answer, based on this dataset, is no.

Across ten production-grade applications built on the most widely used AI development platform in the world, Aikido Security, operating with full source code access at $100 per assessment and averaging 69.7 minutes per scan, identified 47% of critical vulnerabilities. Vector, operating black-box without a single line of source code at an average of $14.31 per assessment and 34.7 minutes per scan, identified 94% of the same critical vulnerabilities. The cost per weighted severity point was $3.73 for Aikido and $0.36 for Vector, a more than 10x efficiency gap in favor of the cheaper, faster, more constrained tool.

The critical findings Vector uniquely identified were not obscure. They included private cryptocurrency keys readable without authentication, an unauthenticated admin panel returning full platform data to any browser, and a four-vulnerability exploit chain on a live NFT platform demonstrating financial manipulation of royalty distributions. Aikido had the source code for all of these applications and did not find them.

This is not an indictment of Aikido as a security company. Aikido offers a broad security platform extending well beyond agentic pentesting: dependency scanning, SAST, secret detection, cloud misconfiguration monitoring, and more. Their Lovable integration is one product within a much larger suite, and their overall platform has genuine value for development teams. This study includes sites where Aikido outperformed Vector on finding count, and those results are presented unchanged.

The point is specific: a pentesting tool with full source code access and $84 million in funding is missing critical findings that a black-box scanner operating at 14% of the cost is not. On the vulnerabilities that matter most, the ones a real attacker finds first, the data is unambiguous.

The era of AI-assisted building has only just begun. The attack surface it generates is real, it is growing, and this paper has documented exactly what it looks like across ten live production targets. The tooling built to protect that surface needs to match its scale, its speed, and its economics.

Appendix A: Full Normalized Finding Tables

A.1 nexusxhub.online (Social Platform)

Finding	Root Cause	Tool	CVSS	Band
Unauthenticated AI edge functions	Missing authentication	Both	9.8	Critical
RLS disabled on profiles: all user PII readable	Broken access control	Vector	9.1	Critical
XSS via LLM output rendering	AI-specific	Vector	8.2	High
Absent rate limiting on auth endpoints	Missing rate limiting	Vector	7.5	High
Prompt injection in AI edge functions	AI-specific	Vector	7.3	High
CORS reflects arbitrary origins on auth endpoint	Misconfiguration	Vector	6.8	Medium
Weak password policy	Auth misconfiguration	Vector	6.5	Medium
User enumeration via signup errors	Information disclosure	Vector	5.3	Medium
Missing CSP and security headers	Misconfiguration	Vector	4.3	Medium
Schema enumeration via PostgREST error verbosity	Information disclosure	Vector	3.7	Low
Hardcoded anon key + JWT in localStorage	Credential exposure	Vector	3.1	Low

A.2 pumpbet.site (Crypto Betting Exchange)

Finding	Root Cause	Tool	CVSS	Band
Solana private keys exposed unauthenticated	Credential exposure	Vector	10.0	Critical
RLS disabled across all tables	Broken access control	Vector	9.8	Critical
Unauthenticated admin function + hardcoded creds	Missing authentication	Both	9.6	Critical
Unauthenticated market, bet, and payout functions	Missing authentication	Both	9.4	Critical
IDOR: authenticated read and write on any user profile	Broken access control	Vector	8.8	High
Prompt injection + full system prompt extraction	AI-specific	Vector	8.4	High
Unauthenticated market settlement trigger	Business logic flaw	Aikido	8.3	High
Unauthenticated Supabase REST: market creation and deletion	Missing authentication	Aikido	8.2	High
Race condition: unlimited credit minting	Business logic flaw	Aikido	8.1	High
CORS wildcard with credentials on auth endpoint	Misconfiguration	Vector	7.8	High
No rate limiting on login endpoint	Missing rate limiting	Vector	7.5	High
No rate limiting on signup endpoint	Missing rate limiting	Vector	7.3	High
Weak password policy	Auth misconfiguration	Both	6.5	Medium
Reusable refresh tokens	Auth misconfiguration	Vector	6.2	Medium
Unauthenticated ai-assistant: arbitrary AI credit consumption	AI-specific	Both	5.3	Medium
Email verification disabled	Auth misconfiguration	Vector	5.8	Medium
Missing security headers, cookie flags, and TLS	Misconfiguration	Both	3.8	Low

A.3 easyhouse.casa (Real Estate Marketplace)

Finding	Root Cause	Tool	CVSS	Band
RLS missing on profiles UPDATE: any user overwrites	Broken access control	Vector	9.3	Critical
RLS missing on profiles SELECT: all profiles readable	Broken access control	Both	8.8	High
Unauthenticated AI edge function: credit drain	AI-specific	Aikido	8.1	High
Absent rate limiting on auth endpoints	Missing rate limiting	Both	7.5	High
Admin endpoints and anon JWT in JS bundle	Credential exposure	Vector	7.2	High
CSV formula injection in admin export	Input validation failure	Aikido	7.0	High
Weak password policy	Auth misconfiguration	Both	6.5	Medium
Email verification disabled	Auth misconfiguration	Vector	5.8	Medium
User enumeration via signup errors	Information disclosure	Both	5.3	Medium
Missing CSP and security headers	Misconfiguration	Both	4.3	Medium
Weak TLS cipher suites	Misconfiguration	Aikido	3.5	Low

A.4 kaleidochain.site (NFT Platform)

Finding	Root Cause	Tool	CVSS	Band
Unauthenticated admin panel: full platform data exposed	Missing authentication	Vector	10.0	Critical
Unauthenticated write: INSERT any table, impersonate wallet	Broken access control	Vector	9.9	Critical
All Supabase tables accessible: complete unauthenticated data dump	Broken access control	Vector	9.8	Critical
Sybil attack + vote stuffing chain: royalty manipulation	Business logic flaw	Vector	9.6	Critical
Unauthenticated read access to all Supabase tables	Broken access control	Both	9.1	Critical
Unauthenticated AI edge function: credit drain	AI-specific	Aikido	8.1	High
Non-atomic marketplace purchase flow	Business logic flaw	Aikido	7.8	High
Anon JWT and admin API endpoints in JS bundle	Credential exposure	Vector	7.2	High
Auth cookie missing security flags	Misconfiguration	Aikido	4.8	Medium
Weak TLS cipher suites	Misconfiguration	Aikido	3.7	Low
Missing or misconfigured security headers	Misconfiguration	Aikido	3.1	Low

A.5 forgesflow.online (Project Collaboration)

Finding	Root Cause	Tool	CVSS	Band
Unauthenticated AI endpoint: financial DoS	AI-specific	Both	9.8	Critical
RLS disabled on profiles table	Broken access control	Both	9.1	Critical
RLS disabled on projects table	Broken access control	Both	9.1	Critical
IDOR on project-copilot: any project accessible	Broken access control	Both	8.7	High
Cross-account task IDOR via Kanban API	Broken access control	Aikido	8.4	High
Unauthenticated AI edge function: ai-match and global-chat	AI-specific	Both	8.1	High
Authenticated read access to other users draft projects	Broken access control	Aikido	8.2	High
Client-side only admin access control	Broken access control	Vector	7.8	High
Student bypasses admin approval workflow	Business logic flaw	Aikido	7.6	High
No rate limiting on login endpoint	Missing rate limiting	Vector	7.5	High
No rate limiting on signup endpoint	Missing rate limiting	Vector	7.5	High
No rate limiting on password reset endpoint	Missing rate limiting	Vector	7.5	High
RLS disabled on events table	Broken access control	Vector	7.2	High
IDOR exposes closed marketplace gigs	Broken access control	Aikido	7.0	High
Unauthenticated AI endpoint: prompt injection vulnerability	AI-specific	Vector	6.8	Medium
Event participant cap bypass	Business logic flaw	Aikido	6.8	Medium
Weak password policy	Auth misconfiguration	Both	6.5	Medium
JWT valid post-logout on REST API	Auth misconfiguration	Vector	6.2	Medium
Email verification disabled	Auth misconfiguration	Vector	5.8	Medium
Unauth. Realtime subscription leaks UUIDs	Information disclosure	Aikido	5.4	Medium
Authenticated IDOR: event participant IDs exposed	Broken access control	Aikido	5.4	Medium
Missing CSP and security headers	Misconfiguration	Both	4.3	Medium
GraphQL introspection enabled: full schema disclosure	Information disclosure	Vector	4.1	Medium
PostgREST verbose error messages: internal schema disclosed	Information disclosure	Vector	3.7	Low
Anon JWT in JS bundle, expires 2090	Credential exposure	Vector	3.8	Low
Weak TLS cipher suites	Misconfiguration	Aikido	3.5	Low

A.6 learnchain.site (Ed-tech / Blockchain Learning)

Finding	Root Cause	Tool	CVSS	Band
Unauthenticated AI chat assistant edge function	AI-specific	Both	8.1	High
No rate limiting on login endpoint	Missing rate limiting	Vector	7.5	High
PostgREST RLS coverage unverifiable: database availability risk	Broken access control	Vector	6.8	Medium
Email verification disabled	Auth misconfiguration	Vector	5.8	Medium
Profile enumeration via Supabase REST profiles endpoint	Information disclosure	Aikido	5.3	Medium
Missing CSP and security headers	Misconfiguration	Both	4.3	Medium
Verbose REST errors: database schema disclosure	Information disclosure	Aikido	4.3	Medium
Auth configuration disclosed via unauthenticated settings endpoint	Information disclosure	Vector	3.7	Low
verify-nft-certificate timing inconsistency	Misconfiguration	Vector	3.7	Low
Supabase anon key in public JavaScript bundle	Credential exposure	Vector	3.1	Low
Weak TLS cipher suites	Misconfiguration	Aikido	3.1	Low

A.7 invoicechain.site (Multi-Tenant Invoicing SaaS)

Finding	Root Cause	Tool	CVSS	Band
Cross-tenant workspace takeover via mutable profiles.company_id	Broken access control	Aikido	9.1	Critical
Cross-tenant invoice disclosure via profiles.email impersonation	Broken access control	Aikido	8.5	High
Invoice status manipulation via unauthorized PATCH	Broken access control	Aikido	7.5	High
Unauthenticated privileged invoice email retry worker	Missing authentication	Aikido	7.3	High
No rate limiting on login endpoint	Missing rate limiting	Vector	7.5	High
Supabase anon JWT key in public JavaScript bundle	Credential exposure	Vector	7.2	High
Invoice recipients access owner-only delivery audit data	Information disclosure	Aikido	5.3	Medium
No rate limiting on password reset and OTP end-points	Missing rate limiting	Vector	4.3	Medium
Missing X-Frame-Options and CSP headers	Misconfiguration	Both	4.3	Medium
lovable-auth cookie missing HttpOnly, Secure, and SameSite	Auth misconfiguration	Aikido	3.1	Low
Weak TLS cipher suites (deduped)	Misconfiguration	Aikido	3.7	Low
__dpl session cookie missing HttpOnly	Misconfiguration	Vector	3.1	Low

A.8 medicarebook.site (Healthcare Appointment Booking)

Finding	Root Cause	Tool	CVSS	Band
MediBot XSS via AI chat (allowDangerousHtml)	AI-specific	Vector	9.3	Critical
Missing auth on medibook-chat (credit exhaustion observed)	AI-specific	Both	8.7	High
Missing auth on appointment-ai edge function	Missing authentication	Vector	8.1	High
Unauthenticated access to patient appointment data	Broken access control	Vector	8.1	High
Redirect_to accepts arbitrary URLs	Broken access control	Vector	6.8	Medium
Role escalation via PostgREST write to user_roles	Broken access control	Vector	6.5	Medium
RPC enumeration: has_role function discoverable	Information disclosure	Vector	5.4	Medium
PostgREST schema leakage via error messages	Information disclosure	Vector	5.3	Medium
Absent rate limiting on auth endpoints	Missing rate limiting	Vector	5.3	Medium
Missing X-Frame-Options on /auth pages	Misconfiguration	Vector	4.3	Medium
Missing CSP header	Misconfiguration	Both	4.3	Medium
Anon JWT exposed in JS bundle	Credential exposure	Vector	3.1	Low
Account enumeration via signup endpoint	Information disclosure	Aikido	3.7	Low
Weak TLS cipher suites	Misconfiguration	Aikido	3.5	Low

A.9 bountyboard.site (Bounty / Task Management)

Finding	Root Cause	Tool	CVSS	Band
XSS via allowDangerousHtml in bounty detail rendering	AI-specific	Vector	7.6	High
Task field manipulation: any task overwriteable	Broken access control	Aikido	7.5	High
Profile enumeration via direct user_id query	Broken access control	Aikido	7.3	High
Cross-user submission reads: unguarded endpoint	Broken access control	Aikido	7.1	High
Cross-user AI conversation poisoning via ai_messages write	Broken access control	Aikido	7.1	High
No rate limiting on signup endpoint	Missing rate limiting	Vector	6.5	Medium
Weak brute-force protection on login endpoint	Missing rate limiting	Vector	6.5	Medium
OTP / email enumeration via GoTrue resend	Information disclosure	Both	5.3	Medium
Contributor-controlled review state on submission creation	Broken access control	Aikido	6.5	Medium
Cross-user activity log read access	Information disclosure	Aikido	6.5	Medium
Single-active-claim control bypass in task workflow	Business logic flaw	Aikido	6.5	Medium
Race condition in ai-chat hourly limiter	Business logic flaw	Aikido	6.5	Medium
No rate limiting on password reset endpoint	Missing rate limiting	Vector	5.3	Medium
Missing X-Frame-Options header enables clickjacking	Misconfiguration	Vector	6.1	Medium
PostgREST data accessible with anon key (RLS enforced)	Information disclosure	Vector	5.3	Medium
Supabase anon JWT hardcoded in public JavaScript bundle	Credential exposure	Vector	5.3	Medium
PostgREST schema enumeration via error messages	Information disclosure	Vector	5.3	Medium
Weak TLS cipher suites (deduped)	Misconfiguration	Aikido	3.7	Low
Missing or misconfigured security headers	Misconfiguration	Aikido	3.1	Low

A.10 trustcrm.site (CRM / AI Chatbot)

Finding	Root Cause	Tool	CVSS	Band
No rate limiting: login + password reset	Missing rate limiting	Vector	8.2	High
Wildcard CORS on PostgREST API	Misconfiguration	Vector	8.2	High
Unauthenticated send-push-reminders edge function	Missing authentication	Aikido	7.3	High
User enumeration via signup endpoint	Information disclosure	Vector	5.3	Medium
Missing CSP / security headers	Misconfiguration	Both	5.3	Medium
PostgREST RLS leaks internal function names	Information disclosure	Both	4.3	Medium
crm-chat: edge function leaks internal error	Information disclosure	Both	4.3	Medium
Weak TLS cipher suites (deduped)	Misconfiguration	Aikido	3.7	Low
__dpl cookie missing HttpOnly	Misconfiguration	Vector	3.1	Low

Appendix B: Severity Weighting Formula

CVSS Band	Score Range	Weight
Critical	9.0 to 10.0	10 points
High	7.0 to 8.9	5 points
Medium	4.0 to 6.9	2 points
Low	0.1 to 3.9	1 point

Vector weighted score: (16 x 10) + (29 x 5) + (42 x 2) + (11 x 1) = 160 + 145 + 84 + 11 = 400.

Aikido weighted score: (8 x 10) + (25 x 5) + (25 x 2) + (13 x 1) = 80 + 125 + 50 + 13 = 268.

Appendix C: Root Cause Taxonomy

Root Cause Class	Vector Findings	Aikido Findings
Broken access control	17	17
Missing authentication	4	5
Missing rate limiting	15	1
Business logic flaw	1	7
Input validation failure / Injection	0	1
Auth misconfiguration	10	4
Credential exposure	9	0
Information disclosure	14	10
Misconfiguration	16	18
AI-specific (prompt injection, LLM rendering, credit drain)	12	8
Total	98	71

Appendix D: CVSS v3.1 Scoring Rationale

Attack Vector. Set to Network for all findings, as every vulnerability in this dataset is exploitable remotely via standard HTTP requests.

Attack Complexity. Set to Low for findings exploitable with straightforward API calls or browser navigation, and High for findings requiring specific preconditions such as concurrent timing for race conditions.

Privileges Required. Set to None for unauthenticated findings, Low for findings requiring any authenticated session, and High for findings requiring elevated role access.

User Interaction. Set to None for all server-side and API findings. Set to Required only for client-side findings where a victim must take an action.

Scope. Set to Unchanged for findings contained within the application boundary and Changed for findings with cross-application or cross-user impact such as stored XSS and sybil chains.

CIA Impacts. Scored High, Low, or None based on the demonstrated or directly implied consequence of successful exploitation in each case.