THE VULNERABILITY DEFICIT

Why Remediation Cannot Outrun Discovery

An Executive Perspective

AVIATRIX®

Vulnerability management, the practice of discovering, prioritizing, and remediating software flaws, has been the foundation of enterprise cybersecurity for two decades. This paper argues that it is mathematically incapable of serving as the primary defense against the current and emerging threat landscape. The argument does not rest on forecasts. It rests on data that already exists and relationships that are already observable.

The Model

The security posture of any organization at any moment can be described by the stock of exploitable, unpatched vulnerabilities in its environment. Call this V. Two forces govern how V changes. New vulnerabilities enter the system through discovery and through new code. Vulnerabilities leave through remediation.

$$V(t) = V(t-1) + D(t) - R(t)$$

D is the rate at which new exploitable vulnerabilities enter the environment. R is the rate at which they are remediated. If D consistently exceeds R, the stock grows without bound. Vulnerability management is the bet that R can be made to exceed D, or at least keep pace. The evidence shows this bet has already been lost, and that structural forces are widening the deficit.

V is not an undifferentiated quantity. Vulnerabilities vary in severity, exploitability, and exposure. A growing V composed of low-severity, unreachable flaws is categorically different from a growing V composed of critical, internet-facing weaknesses. The argument that follows depends on the rate of consequential vulnerability accumulation, the subset that is exploitable, reachable, and material to the business, exceeding the rate at which that subset can be remediated. The evidence shows this is already occurring.

The Discovery Rate Is Compounding

D is the sum of three independently growing inputs.

AI discovery capability. The constraint on vulnerability discovery has always been human attention and expertise. That constraint has been removed. Frontier models now autonomously discover exploitable zero-days in commercial software at rates that were impossible twelve months ago. Anthropic's Claude Mythos model, evaluated in April 2026, autonomously found 181 exploitable zero-day vulnerabilities in Firefox where the previous generation found two. Days later, Z.ai released GLM-5.1, a 754 billion parameter open-weight model under an MIT license with no safety guardrails, closing a meaningful share of the capability gap between frontier labs and freely distributed models.

A single vendor's benchmark does not prove a permanent compound trajectory. What matters is the nature of the progression. Model capability curves have moved on steep, consistent slopes for three years across every frontier lab, and open-source models have narrowed the gap with frontier models on every major benchmark as soon as weights become available. The discovery constraint is now bounded by compute allocation and model capability, both increasing rapidly, and by the finite attack surface of any given codebase. Whether D compounds at 10x per generation or 90x per generation does not change the structural argument. It changes the timeline. The industry should internalize the shape of the curve, not the precise slope at any single data point.

Codebase growth. The global codebase is not a fixed body of work being audited for historical flaws. It is expanding at an accelerating rate, and AI-assisted development is a primary driver of that acceleration. Every new function creates interaction surfaces with existing code. The vulnerability surface grows faster than linearly with code volume, because the interaction space between components expands combinatorially. AI is simultaneously writing the code and finding the flaws in all code ever written. Both inputs to D are growing.

Dependency chain amplification. Modern applications are approximately 80 to 90% third-party code. The average enterprise application pulls in hundreds of open-source libraries, each with its own transitive dependency tree often five to seven levels deep. A vulnerability at any level is a vulnerability in the application. Log4Shell demonstrated this. A single flaw in a logging library three levels deep in dependency chains exposed virtually every Java application on earth. The vulnerability surface is the transitive closure of every dependency the code touches, aggregating vulnerabilities from thousands of independent development teams worldwide.

The Remediation Rate Has a Ceiling

The most important empirical evidence in this paper comes from CISA's Known Exploited Vulnerabilities program, analyzed by the Qualys Threat Research Unit across 1.1 billion remediation records from more than 10,000 organizations over four years. Two findings.

Organizations increased remediation effort by a factor of 6.5 in a single year. They closed 6.5 times more tickets. The percentage of critical vulnerabilities still unresolved at seven days worsened from 56% to 63%.

The KEV catalog is a curated list of vulnerabilities known to be actively exploited, and the Qualys dataset skews toward federal agencies and critical infrastructure. The remediation ceiling observed in this population may be higher for cloud-native companies with modern deployment pipelines. The qualification strengthens the argument rather than weakens it. The KEV catalog tracks the vulnerabilities that actually determine whether an organization gets breached, the ones attackers are actively exploiting. If the ceiling exists for these, it is the ceiling that matters. And the 6.5x-effort-for-negative-return finding is the empirical signature of a system approaching its asymptotic limit: massive effort, negative marginal returns.

R has an upper bound, Rmax, determined by the sequential, judgment-intensive nature of safely changing complex production systems. Identify the vulnerability. Write a patch. Test it against every dependent system. Schedule a deployment window. Deploy without breaking production. Verify the fix. Each step involves human judgment and cross-team coordination that does not compress linearly with additional resources.

Modern cloud-native architectures do compress remediation timelines for a meaningful subset of workloads. Immutable infrastructure eliminates the risk of patching running systems. GitOps pipelines with canary analysis and instant rollback compress the testing-deployment-verification sequence from weeks to hours. Serverless architectures eliminate the concept of patching a running system entirely.

For organizations that have fully adopted these practices, Rmax is higher. The question is whether it is high enough to match D. Three factors suggest not. First, even in fully automated pipelines, the bottleneck is not deployment.

It is the upstream work of identifying which vulnerabilities are exploitable in a specific configuration, prioritizing across hundreds of simultaneous findings, and validating that fixes do not break dependent services. Automation compresses the deployment step, which is 10 to 15% of the total timeline. The remaining 85 to 90% remains organizational. Second, even the most modern enterprises run heterogeneous environments. Cloud-native workloads coexist with legacy systems, third-party SaaS, and managed services where the customer cannot patch at all. Rmax for the environment is determined by the slowest segment, not the fastest. Third, a 10x improvement in remediation speed does not change the structural math when D has increased by orders of magnitude. A higher ceiling is still a ceiling, and the forces driving D operate on exponential curves while improvements in R are linear.

Iatrogenic Risk

Research consistently shows that 5 to 15% of security patches introduce new defects, a subset of which are security-relevant. Remediation is not purely subtractive. Some fraction of R feeds back into D.

At 1,000 remediations per month with a 5% regression rate, approximately 50 new defects are introduced, of which perhaps 10 to 20 are security-relevant. At 10,000 remediations per month, the scale implied by the 6.5x effort increase, the iatrogenic contribution is 100 to 200 new security-relevant defects. This does not dominate D. It does mean that increasing remediation volume carries a cost that further constrains the net effectiveness of R. At the extreme, remediation encounters a point of negative marginal returns. The CISA data suggests some organizations may already be approaching it.

The Complete Model

Incorporating all material factors, the full relationship is:

$$V(t) = V(t-1) + D(t, C(t)) - Reff(t) + f(R(t)) + M(t)$$

D is the discovery rate, a function of AI capability and the size of the global codebase C(t), searched across the full depth of every transitive dependency chain. Reff is effective remediation, bounded by the ceiling of organizational execution. f(R) is iatrogenic feedback, the fraction of patches that create new vulnerabilities. M(t) represents the vulnerability surface that exists entirely outside the domain of patching, including misconfigurations, architectural design choices such as flat networks, and overpermissioned non-human identities. These cannot be remediated through code. They require structural change.

D is a function of independently growing inputs applied to an expanding surface. Reff is bounded by a ceiling that can be raised but not eliminated. f feeds back positively. M grows independently. Under current conditions and foreseeable trajectories, V(t) diverges. This system will not be brought into equilibrium by increasing R, regardless of the resources applied.

This is not a formal proof of divergence under all parameter values. It is a first-principles analysis showing that the structural forces driving D and constraining R operate in directions and at magnitudes that make convergence implausible.

The Collapse of the Exploitation Window

The model above assumes defenders learn about vulnerabilities in time to act. For the vulnerabilities that matter most, that assumption is failing.

The broad exploitation window has compressed for two decades, from 771 days in 2018 to 84 days in 2021 to 6 days in 2023 to 4 hours in 2024. For vulnerabilities that reach the CISA Known Exploited list, the window has inverted. Exploitation is observed before the vulnerability appears in any public database, because the catalog is populated after exploitation is detected. The selection bias is real, and the mechanism it reveals is not an artifact. Attackers discover and weaponize before defenders are told. For the class of vulnerabilities that actually determines whether an organization suffers a material breach, the foundational sequence of vulnerability management (discover, disclose, patch, deploy) is already broken at step one.

The bulk of CVEs, the thousands of moderate-severity vulnerabilities disclosed each year, still have exploitation windows measured in weeks or months, and remediation remains viable for managing them. The collapse described here applies to high-value zero-days and actively exploited vulnerabilities. These are the vulnerabilities that determine whether an organization suffers a material breach. The window for the vulnerabilities that matter most is the one that has collapsed. And 82% of intrusions now use valid credentials through legitimate channels, producing no anomalous signal at all.

The Structural Asymmetry

The attacker needs to find and exploit one exploitable vulnerability that reaches something valuable. The defender relying primarily on remediation needs to find and patch all such vulnerabilities before they are exploited. This asymmetry worsens as V grows. The attacker's problem is bounded by the existence of a single reachable exploit. The defender's problem scales with the total consequential surface.

The asymmetry is compounded by the nature of the work. Discovery is a search problem, and AI excels at search. Remediation is a transformation problem, safely changing production systems in complex, interdependent environments where every change carries risk. Discovery scales with compute. Remediation scales with organizational complexity. These are fundamentally different curves. The bottleneck in remediation is not writing the fix. It is deploying it safely. That bottleneck is organizational, not computational, and no amount of compute eliminates it.

This asymmetry is specific to remediation as a strategy. A defense strategy based on containment changes the asymmetry by reducing the set of reachable vulnerabilities regardless of whether they are patched. If a compromised workload can only communicate with three other services and one external destination, the attacker's lateral movement and exfiltration options collapse from thousands to four, whether or not the vulnerability that enabled the initial compromise has been patched. Containment does not eliminate the asymmetry between attackers and defenders. It shifts the asymmetry from the remediation domain, where it is structurally unwinnable, to the architectural domain, where the defender has the advantage.

Why the Counterarguments Are Insufficient

AI-assisted remediation. AI can write patches and suggest fixes, but writing the fix is 10 to 15% of the enterprise remediation timeline. The remaining 85 to 90% (impact assessment, testing, change approval, deployment, verification) is organizational work. Even a 10x improvement in fix-writing speed produces less than 2x improvement in end-to-end remediation time. That does not change the math when D has increased by orders of magnitude. And if AI eventually automates the organizational loop as well, the argument for containment only strengthens, because the time-independence of architectural defense compounds with every improvement elsewhere.

Reducing the vulnerability density of new code. Memory-safe languages, secure-by-design practices, and formal verification each reduce D for a subset of new code. Memory-safe languages eliminate memory corruption bugs, which account for approximately 70% of critical CVEs in systems software. Formal verification can mathematically prove correctness for small, critical components. All three approaches share structural limitations. They do not affect the hundreds of billions of lines already in production. AI-assisted development produces code with vulnerability rates comparable to or slightly worse than human-written code. And even flawless new code inherits the vulnerabilities of every library it imports. These are valuable practices. None changes the relationship between D and R.

The Implication

Vulnerability management is not dead. Patching remains the correct response to the bulk of known vulnerabilities, particularly the moderate-severity findings that constitute most of any organization's inventory. Patching is hygiene. Discipline in hygiene matters.

Vulnerability management is no longer viable as the primary defense strategy, the strategy an organization relies on to prevent material breach. The discovery rate is compounding on multiple curves. The remediation rate has a ceiling. The feedback loops are positive. The exploitation window has collapsed for the vulnerabilities that matter most. And the attacker's structural advantage worsens as the deficit grows.

The only rational response to a system where breach is structurally inevitable is to build architecture that makes each breach survivable.

When prevention fails and detection is too slow, containment decides whether the incident becomes a breach.

About Aviatrix

Aviatrix® is pioneering the Cloud Native Security Fabric – the architecture the Containment Era requires. The Cloud Native Security Fabric governs every workload communication path across every cloud, every VPC, every Kubernetes cluster, and every serverless function, from a single policy plane. One rule. Universal propagation. Enforced at the workload, not at a chokepoint. Trusted by more than 500 of the world's leading enterprises. For more information, visit aviatrix.ai.

Notes

[1] Anthropic Red Team Disclosure, April 7, 2026. Claude Mythos model evaluation against browser and operating system targets. Z.ai GLM-5.1 open-weight release, April 7, 2026.
[2] GitHub Innovation Graph and Octoverse Reports, 2024. AI-assisted code contributions across the GitHub platform.
[3] Synopsys Open Source Security and Risk Analysis (OSSRA) Report, 2024. 96% of commercial codebases contain open-source components; average application includes 500+ open-source dependencies.
[4] Qualys Threat Research Unit, analysis of CISA Known Exploited Vulnerabilities catalog. Dataset: 1.1 billion remediation records across 10,000+ organizations, 2020–2024.
[5] Brooks, Frederick P. The Mythical Man-Month (1975). Communication overhead grows as n(n-1)/2.
[6] Enterprise remediation timeline analysis based on Gartner and Forrester industry benchmarks.
[7] Perry and Stieg (2000), Yin et al. (2011), and Microsoft internal studies on patch regression rates.
[8] Qualys Threat Research Unit, analysis of CISA Known Exploited Vulnerabilities catalog. Median time-to-exploit for KEV-listed vulnerabilities.
[9] Sergej Epp, CISO of Sysdig. Time-to-exploit trend analysis compiled from NVD, Mandiant, and CISA data, March 2026.
[10] CrowdStrike 2026 Global Threat Report. 82% of intrusions used identity-based techniques with no malware. Average breakout time: 29 minutes.
[11] Microsoft Security Response Center and Google Project Zero reports on memory safety. White House ONCD statement, February 2024.
[12] Stanford University, “Do Users Write More Insecure Code with AI Assistants?” (2023).
[13] SANS Institute and Cloud Security Alliance, “Project Mythos: AI and the New Threat Landscape,” April 15, 2026. 60+ authors, 80+ CISO reviewers.