Tuskira™

Tuskira Research Report

The Emerging Patch Gap

What Anthropic Mythos Data Reveals About AI-Driven Vulnerability Discovery and Enterprise Remediation

Based on analysis of 1,596 verified vulnerabilities disclosed across 281 open-source projects over 63 days | June 2026

Executive Summary

AI-driven vulnerability discovery has crossed a threshold enterprise security programs were not built to absorb.

Between March 20 and May 22, 2026, Anthropic's Claude Mythos Preview analyzed more than 23,000 open-source code paths and disclosed 1,596 verified vulnerabilities across 281 distinct open-source projects. The program operates through a responsible coordinated vulnerability disclosure process: findings are triaged, privately reported to maintainers, patched when possible, and published only after disclosure windows close.

That process is working as designed. The problem is what happens downstream.

For defenders, there is now a structural timing gap between the moment a vulnerability is privately disclosed to a maintainer and the moment enterprise security teams can see, prioritize, validate, and deploy a fix. Conventional vulnerability management tools usually become useful only after an advisory is public, after NVD and ecosystem feeds ingest it, and after commercial scanners refresh. Enterprise patch deployment then adds another validation window, often measured in weeks.

Tuskira Research calls this window **the Patch Gap.**

The central finding of this report is that AI-driven vulnerability discovery is now outpacing remediation by approximately one order of magnitude or more. In the May 22, 2026 snapshot analyzed for this report, Mythos-generated disclosures were created at a rate of approximately 25.3 verified vulnerabilities per day, while Mythos-attributed remediation occurred at approximately 1.5 vulnerabilities per day. That implies a discovery-to-remediation ratio of approximately

16.5x and a net deficit growth rate of approximately 23.8 unaddressed disclosures per day.

16.5x and a net deficit growth rate of approximately 23.8 unaddressed disclosures per day.

Core Research Finding

AI-driven vulnerability discovery is outpacing remediation by approximately 16.5x in the analyzed snapshot.

Metric	Observed Rate Per Day
Verified vulnerabilities discovered	25.3
Verified vulnerabilities remediated	1.5
Net deficit growth	23.8

The Patch Gap is not a critique of coordinated vulnerability disclosure. Responsible disclosure remains necessary.

Implication: Vulnerability discovery is accelerating faster than maintainers and enterprises can safely remediate and deploy fixes, creating a growing vulnerability deficit.

This does not mean every unremediated disclosure is exploitable in every enterprise environment. Most vulnerabilities do not map to reachable, exposed, business-critical runtime paths in any given organization. But it does mean enterprises can no longer rely on a simple scan-find-patch model. They need a decision model that works before the CVE feed fires and during the weeks when a patch exists but is not yet safely deployed.

Tuskira Research proposes a four-pillar operating model for defending inside the Patch Gap:

1. **Reachability:** Is the vulnerable code path actually exercised in production?
2. **Exposure:** Who or what can reach the vulnerable instance, and what is the blast radius?
3. **Active exploitation:** Is there evidence of exploitation or an indicator of attack in the environment?
4. **Defense coverage:** Do existing controls block the specific exploit pattern, and has that coverage been validated?

Together, these inputs allow security leaders to make defensible patch decisions: emergency patch, staged patch, compensate and monitor, or defer with evidence.

The Patch Gap is not a critique of coordinated vulnerability disclosure. Responsible disclosure remains necessary. The issue is that the time disclosure necessarily takes is now being filled by a much faster discovery engine.

Enterprises need to operate at discovery cadence, not only remediation cadence.

The Five Findings

Finding 1: AI-driven vulnerability discovery is outpacing public advisory visibility.

In the analyzed snapshot, Claude Mythos Preview disclosed 1,596 verified vulnerabilities across 281 open-source projects in 63 days.

The published data indicates:

Measure	Snapshot Value
Total disclosed vulnerabilities	1,596
Distinct open-source projects	281
Observation window	63 days
Approximate disclosure rate	25.3/day
Partner-firm-verified true-positive rate	90.8%

This is not a normal increase in vulnerability discovery volume. The observed cadence is materially above prior public AI-driven discovery programs and represents a shift from isolated breakthroughs to sustained, high-volume disclosure.

The important market implication is not simply that more bugs are being found. The important implication is that discovery has moved faster than the downstream systems enterprises use to consume, score, patch, and validate those discoveries.

Why This Is Different

The Mythos snapshot is not directly comparable to every prior vulnerability-discovery program. Each program has a different scope, triage process, disclosure policy, and operating model. But the order-of-magnitude signal is difficult to ignore.

Program	Public Window	Published Output
Google Big Sleep	First year of public operation	Approximately 20 OSS findings plus one wild zero-day
DARPA AIxCC Finals	Tournament burst	18 real vulnerabilities, 11 patched
OSS-Fuzz-Gen	November 2024 snapshot	26 vulnerabilities across 272 C++ projects
OSS-Fuzz	2016-2025 lifetime	More than 13,000 vulnerabilities
Claude Mythos Preview	First 63 dayz	1,596 disclosed vulnerabilities

The cleanest comparison is not any single ratio. It is the shift in shape: Mythos reached roughly 12% of OSS-Fuzz's nine-year public vulnerability corpus in about 2% of the elapsed time, while operating through a responsible disclosure pipeline with external security-firm triage.

There are also important counterweights. Anthropic describes true positives as one proxy for impact, not the whole impact story. Severity grading is imperfect: the source report found 58.7% exact-severity agreement and 94.4% within-one-severity agreement between Claude and vendor severity, with Claude tending to over-call criticality.

Maintainer experiences vary as well; some projects have reported low-severity or false-positive submissions within the broader aggregate.

That mix strengthens the conclusion rather than weakening it. The story is not "AI found 1,596 equally urgent bugs." The story is that AI-driven discovery has reached a sustained volume that makes old prioritization assumptions unsafe. Security teams need runtime evidence to separate reachable, exposed, exploitable risk from noise, duplicates,low-impact findings, and issues outside a project's threat model.

Finding 2: Most disclosed vulnerabilities were not visible through public advisories at the snapshot.

Approximately 95% of Mythos disclosures had no public advisory at the May 22, 2026 snapshot.

Pipeline State	Count	Share
Total disclosed to maintainers	1,596	100%
Acknowledged by maintainers	1,451	90.9%
Patched upstream in response	97	6.1%
Publicly identifiable advisory records	26 records / 32 identifiers	1.6%-2.0%
Not yet patched	1,499	93.9%

The low public-advisory visibility is not a failure of responsible disclosure. It is the result of responsible disclosure operating correctly. Maintainers need time to review, patch, coordinate, and publish. Advisory databases need time to ingest. Commercial tools need time to update. Enterprises need time to validate and deploy.

This creates the Patch Gap: the period after a vulnerability is known upstream but before normal enterprise tooling can fully act on it.

For a typical disclosure, the end-to-end window from private disclosure to deployed enterprise patch can be structurally measured in the range of 90 to 150 days, with the largest contributors being the coordinated disclosure window and the enterprise validation window.

Finding 3: Maintainer acknowledgment does not mean patch availability.

Maintainers responded quickly. The median acknowledgment time was 0.2 days.

But fast acknowledgment did not translate into fast remediation. Only 97 of 1,596 disclosures were marked as patched in response at the snapshot, or 6.1%.

That distinction matters. Security programs often treat acknowledgment, advisory publication, scanner detection, and patch deployment as if they are steps in a smooth queue. In practice, they are separate bottlenecks.

Maintainers may acknowledge a valid issue but still need time to:

- ✓ Understand the root cause.
- ✓ Design a fix that does not break legitimate behavior.
- ✓ Coordinate with downstream package maintainers.
- ✓ Decide whether the issue fits the project threat model.
- ✓ Handle multiple AI-submitted findings at once.

The measured 6.1% patched-in-response rate should be treated as a lower bound, not an exact final remediation rate. Some fixes may be shipped quietly or without attribution to Mythos. Even so, the snapshot shows a clear mismatch between discovery velocity and visible remediation velocity.

Finding 4: Enterprise patch deployment creates a second exposure window.

Even after an upstream patch exists, enterprises usually cannot deploy it everywhere immediately. For non-trivial dependencies, a patch can introduce regression risk:

Regression Class	Enterprise Impact
API drift	Valid payloads begin failing after an input validation change.
Behavioral drift	A memory safety fix changes timing or allocation behavior.
Transitive dependency churn	A library upgrade forces peer-version changes.
Performance regression	Additional checks or copies affect latency.
Compliance recertification	Crypto or certified components require additional review.

In many enterprise environments, validation windows commonly run 2 to 6 weeks for language packages and longer for native, FFI-heavy, certified, embedded, or regulated components. During this period, the vulnerability may be public, exploit tooling may update, scanners may alert, and production may remain unpatched. This is the second exposure layer: not pre-CVE invisibility, but post-patch deployment lag.

The operational question becomes: which systems require emergency patching, and which can be safely handled through the validated patch lane? That decision cannot be made from CVSS alone. It requires runtime context.

Finding 5: Package-ecosystem fan-out amplifies downstream alert volume.

Mythos counts upstream disclosures. Enterprise tools consume package-level advisories. One upstream vulnerability can fan out into many downstream alerts across package ecosystems, Linux distributions, container images, language packages, and vendor-maintained variants.

The report's analysis found examples such as:

- A single upstream ImageMagick CVE propagating to 18 or more NuGet package variants through Magick.NET.
- Debian nginx-source indexing 14 CVE-2026 records derived from the program.
- Source-only projects such as wolfSSL, nginx, jq, MapServer, FreeRDP, ImageMagick, libyang, and Mastodon propagating through distribution rebuilds rather than one package feed.

The downstream alert burden will therefore be larger than the upstream vulnerability count suggests. For defenders, the relevant unit is not **"one upstream finding."** It is "**every reachable affected instance in the enterprise runtime."**

The Vulnerability Deficit

The five findings point to one structural pattern: AI-driven discovery is generating verified vulnerabilities faster than the maintainer and enterprise remediation pipeline can consume them.

Tuskira Research calls this mismatch the vulnerability deficit. At the May 22, 2026 snapshot:

Pipeline Flow	Rate	Source
Discovery flow	25.3/day	1,596 disclosures / 63 days
Remediation flow	1.5/day	97 fixed in response / 63 days
Net deficit accumulation	23.8/day	Discovery flow minus remediation flow
Discovery-to-remediation ratio	16.5x	1,596 / 97

The ratio should be interpreted carefully. The remediation figure is a measured lower bound because not every upstream fix is necessarily attributed to Mythos in the published data. A conservative reading is that discovery is outpacing visible remediation by at least an order of magnitude. Even with that caveat, the market implication is significant.

How to Read the 16.5x Finding

The 16.5x figure is not a prediction that the vulnerability backlog will grow forever at the same rate. It is a snapshot measurement of the relationship between two visible flows in the first 63 days of the Mythos disclosure program:

- The rate at which verified vulnerabilities were disclosed to maintainers.
- The rate at which vulnerabilities were marked as fixed in response.

Several forces could change the ratio over time. Maintainer funding, agentic patch-generation tools, disclosure-process changes, and increased project hardening could improve remediation velocity. At the same time, more frontier-model vulnerability discovery programs could increase discovery velocity across the ecosystem.

For launch and media use, the most defensible formulation is:

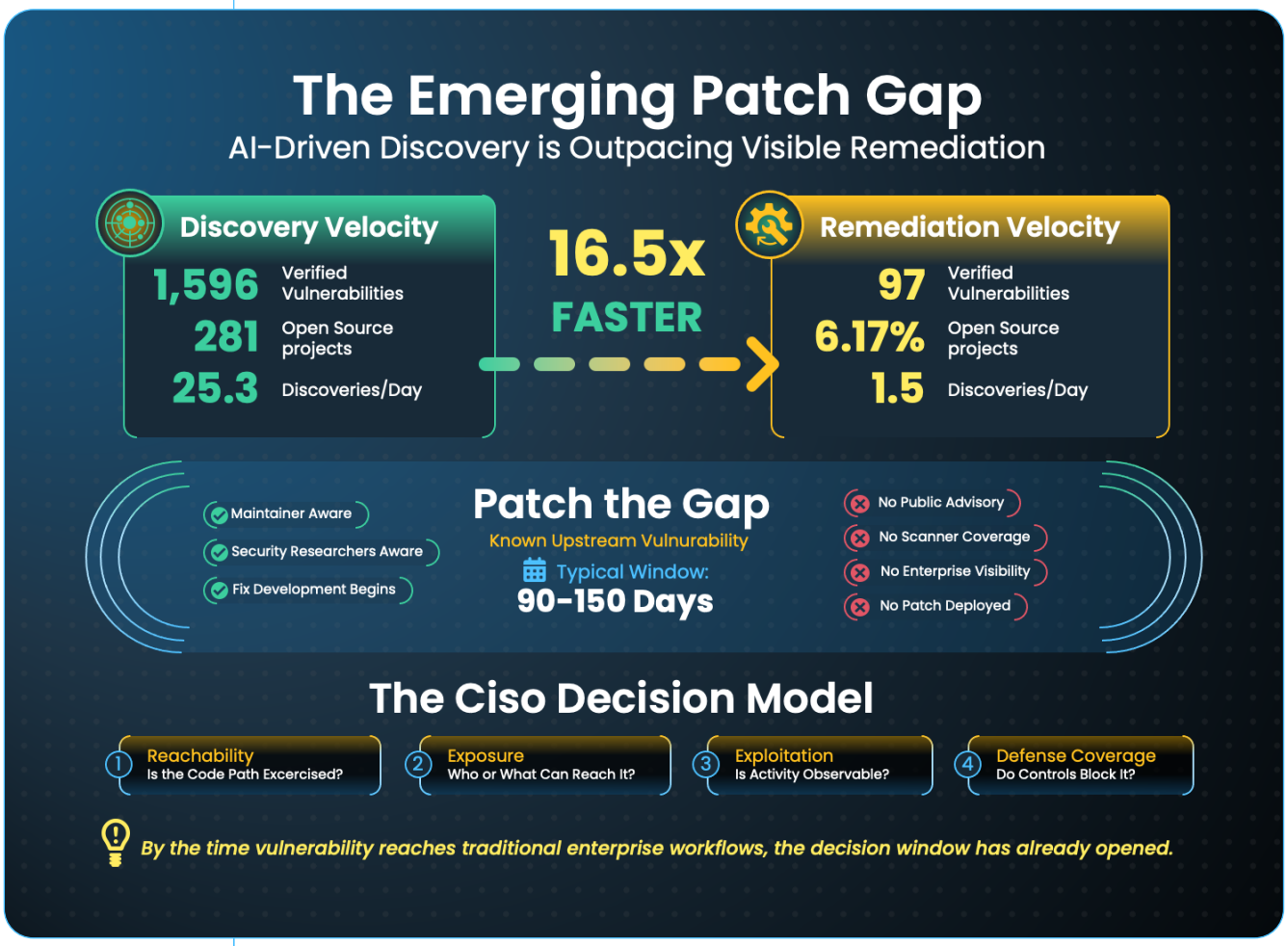
In Tuskira Research's May 22, 2026 snapshot, AI-driven vulnerability discovery outpaced visible Mythos-attributed remediation by approximately 16.5x.

That phrasing is precise, grounded, and hard to dismiss. It avoids implying that every finding remains exploitable forever or that the same ratio applies to every future disclosure program.

When discovery outruns remediation, enterprises cannot patch everything at emergency speed. They also cannot ignore the backlog. They need a way to determine, per disclosure and per runtime instance:

- Whether the vulnerable code path is actually reachable.
- Whether the instance is externally or materially exposed.
- Whether exploitation is active or likely.
- Whether compensating controls block the exploit path.
- Whether the patch should enter an emergency lane or a validated lane.

The vulnerability deficit converts vulnerability management from a list-processing problem into a runtime decision problem.



The research indicates that AI-driven vulnerability discovery is now occurring significantly faster than visible remediation. During this period, vulnerabilities may be known to maintainers and researchers while remaining invisible to enterprise vulnerability management workflows. This creates a Patch Gap: a period where organizations must make risk decisions before traditional advisory-driven processes can respond.

Evidence and Caveats

The claims in this launch report are intentionally snapshot-bound. They are based on Anthropic's public Claude Mythos Preview coordinated vulnerability disclosure data as captured on May 22, 2026 at 17:27:03Z.

Evidence Element	Source or Treatment
Primary dashboard	https://red.anthropic.com/2026/cvd/
Aggregate counters	https://red.anthropic.com/2026/cvd/data/payload.json
Transparency log	https://red.anthropic.com/2026/cvd/data/ledger.json
Snapshot fingerprints	SHA-256 hashes are preserved in Methodology and Sources.

This report separates three kinds of claims:

- **Measured claims:** Counts and rates visible in the May 22 Mythos snapshot, including 1,596 disclosed vulnerabilities, 281 projects, 97 fixed-in-response disclosures, and the 63-day observation window.
- **Derived claims:** Ratios calculated from measured claims, including the 16.5x discovery-to-remediation ratio and the 23.8/day net deficit.
- **Operational claims:** Enterprise implications based on how advisory ingestion, scanner refresh, patch validation, runtime control validation, and CISO risk decisions work in practice.

The most important caveat is remediation attribution. The 97 fixed-in-response figure is a lower bound on actual remediation because some maintainers may ship fixes quietly, merge security-shaped changes without Mythos attribution, or resolve issues outside the visible fixed-in-response field. A conservative reading is therefore not that exactly 93.9% of disclosures remain forever unresolved. It is that visible AI-driven discovery is outpacing visible AI-attributed remediation by approximately an order of magnitude or more.

The second caveat is enterprise applicability. The ecosystem-level deficit is not the same as the emergency patch deficit inside any one company. Most enterprises will not run every affected project, and many affected dependencies will not be reachable or exposed in production. That is why the report's operating model starts with reachability and exposure rather than severity alone.

The third caveat is freshness. Anthropic publishes incremental updates. Any team citing the numbers in real time should refresh the dashboard, payload.json, and ledger.json before quoting the snapshot.

Why This Matters for Enterprises

The CVE feed is now late.

For decades, enterprise vulnerability management has been organized around public identifiers: CVEs, GHSAs, NVD enrichment, SCA alerts, SAST alerts, scanner findings, ticket queues, remediation SLAs, and patch reporting. That model assumes public advisory visibility is close enough to the moment defenders need to act. AI-driven discovery breaks that assumption.

In a high-volume coordinated disclosure environment, a vulnerability can be:

- Discovered by an AI system.Triaged by a security research partner.
- Privately disclosed to a maintainer.
- Acknowledged by the maintainer.
- Patched upstream.
- Held until the disclosure window closes.
- Published as a CVE or GHSA.
- Ingested by advisory databases.
- Ingested by commercial scanners.
- Validated and deployed by the enterprise.

Most enterprise programs begin meaningful prioritization around steps 7 through 9. The risk begins earlier.

Scanner output is necessary but no longer sufficient.

SCA, SAST, CNAPP, ASPM, and vulnerability-management tooling remain necessary. Enterprises still need inventory, advisory ingestion, dependency mapping, ticketing, patch status, and compliance reporting.

But scanner output alone cannot answer the patch decision:

- ✓ **Is this vulnerable dependency actually exercised?**
- ✓ **Is the affected service internet-facing?**
- ✓ **Is the service protected by WAF, network policy, identity gates, or EDR?**
- ✓ **Is the exploit pattern observable?**
- ✓ **Are attackers attempting it now?**
- ✓ **Can the organization safely defer patching without accepting unmanaged risk?**

Those questions require runtime evidence.

The CISO decision is becoming evidence-based, not policy-based.

Traditional policies often route vulnerabilities by severity and SLA: critical in X days, high in Y days, medium in Z days. That model weakens when discovery volume explodes and patch capacity remains finite.

A critical vulnerability in an unreachable code path may not require an emergency patch. A high vulnerability in a public, reachable, unprotected service may require immediate action. A medium vulnerability with active exploitation may be more urgent than a critical vulnerability with strong compensating controls.

The CISO needs a defensible answer to four questions:

1. **Are we exposed?** *Is the vulnerable code path reachable in production?*
2. **Who can reach it?** *What is the identity, network, data, and blast-radius context?*
3. **Is anyone exploiting it?** *Are there indicators of attack in logs, WAF, EDR, runtime telemetry, or network data?*
4. **Are controls actually working?** *Do existing defenses block the exploit pattern, and has that been validated?*

The organizations that can answer these questions will preserve patch capacity, reduce production risk, and make better decisions under disclosure pressure.

The Pre-CVE Signal Model

The Patch Gap does not mean defenders are blind. It means the most useful signal often appears before the traditional CVE-to-scanner pipeline completes.

Tuskira Research identified five layers of pre-CVE signal that security teams can operationalize today.

Layer 1: Transparency-log monitoring

Anthropic's transparency log publishes evidence that disclosures exist before the full details are public. For unrevealed entries, the project name and technical details remain protected, but timing, reveal state, severity fields, bug class fields, and fix-state signals can still provide directional intelligence.

Defender action: Mirror relevant transparency logs internally and treat reveal-tier changes as an event stream. When an entry transitions to revealed, asset-mapping work should already be started.

Layer 2: Bug-class distribution

Even without a public CVE, bug-class trends are actionable. The source report found dominant classes such as heap-buffer-overflow, auth-bypass, broken-access-control, type-confusion, denial-of-service, use-after-free, stack-buffer-overflow, information disclosure, privilege escalation, XSS, SSRF, path traversal, SQL injection, and integer overflow.

Defender action: Harden control coverage against the top bug classes before specific identifiers land. This includes WAF policies, input validation, memory-safety mitigations, egress restrictions, privilege boundaries, and runtime monitoring.

Layer 3: Partner-firm attribution

The Mythos program used external security research firms for triage. When public advisories appear, the advisory may credit the partner firm rather than Anthropic or Mythos directly.

Defender action: Monitor GHSA and advisory feeds for credits associated with known Mythos partner firms. This can help identify Mythos-derived advisories even when the advisory itself does not name Mythos.

Layer 4: Upstream commit monitoring

Fix commits often land before public advisory publication. Commit messages, diffs, tests, and release notes can reveal security-shaped changes such as input validation, memory safety, authorization checks, bounds checks, parser fixes, or unsafe path handling.

Defender action: Monitor upstream commits for runtime-exercised dependencies. This is most valuable when connected to reachability, because no security team can manually monitor every dependency in every package tree.

Layer 5: Runtime reachability and exposure

Reachability and exposure do not require a CVE. A security team can determine which dependencies are present, which code paths are executed, which services are internet-facing, and which identities or tenants can reach them before a specific vulnerability is published.

Defender action: Pre-compute runtime context so that when a CVE, GHSA, or pre-CVE signal appears, the organization already knows whether the affected software is present, reachable, exposed, and business-critical.

The result is a shift in operating posture. Instead of waiting for public advisory ingestion to begin the investigation, defenders can use pre-CVE signals to prepare the decision in advance. When the identifier lands, it becomes confirmation, not a starting gun.

Recommendations

1. Maintain runtime reachability evidence.

Enterprises should maintain continuous evidence of which open-source code paths are exercised by production traffic. This requires joining static analysis, SBOM data, dependency presence, runtime tracing, eBPF telemetry, IAST-style request evidence, and call-graph context. Reachability is CVE-independent. It can be computed before a disclosure exists and applied as soon as a disclosure becomes relevant.

The practical benefit is surface-area reduction. Instead of asking, **"Where do we have this dependency?"** the better question is, **"Where do we exercise the vulnerable path?"**

2. Maintain continuous runtime exposure mapping.

Reachability alone does not determine urgency. The same vulnerable library can exist in:

- A public internet-facing service.
- An internal service reachable by many tenants.
- A private workload reachable only by a narrow service account.
- A development environment with no sensitive data.

Each instance has a different risk profile. Enterprises should maintain a graph that joins workload inventory, dependency versions, identity boundaries, network paths, service mesh policy, data classification, and business criticality. Exposure mapping turns one CVE into a ranked set of instance-level decisions.

3. Operationalize active-exploitation detection ahead of public CVE ingestion.

During the Patch Gap, defenders should push detection signal forward of public advisory visibility. That includes:

- Monitoring upstream security-shaped commits for runtime-exercised dependencies.
- Watching advisory credits and partner-firm attribution signals.
- Translating exploit-path analysis into YARA, Sigma, WAF, eBPF, Falco, EDR, or SIEM detection logic.
- Using indicators of attack, not only indicators of compromise.

The goal is not to replace patching. The goal is to know whether exploitation is occurring while the organization validates and stages the patch.

4. Continuously validate compensating-control coverage.

A compensating control is only useful if it actually blocks the relevant exploit path. Enterprises should test deployed controls against exploit-specific invariants:

- Does the WAF block the malformed request pattern?
- Does network policy restrict post-exploit egress?
- Does EDR detect the process behavior?
- Does identity policy prevent privilege escalation?
- Does runtime policy block the syscall sequence?

Control coverage should be expressed in standards-based language where possible, such as MITRE D3FEND and ATT&CK mappings, so that the result is defensible to auditors, regulators, and the board.

The Patch-Gap Defense Doctrine

The Patch-Gap Defense Doctrine is a runtime decision loop for operating during the period between upstream disclosure and safely deployed enterprise remediation.

It is built on four pillars:

Pillar	Question	Decision Value
Reachability	Is the vulnerable code path exercised?	Eliminates irrelevant emergency work.
Exposure	Who can reach the instance, and what is the blast radius?	Prioritizes high-risk runtime instances.
Active exploitation	Is exploitation occurring or observable?	Escalates from validated patch lane to emergency lane.
Defense coverage	Do controls block the exploit path?	Buys time for safe patch validation.

Decision Loop

1. **A new disclosure signal lands.** It may be a CVE, GHSA, upstream commit, transparency-log transition, or unattributed pre-CVE signal.
2. **Run the reachability check.** If the vulnerable path is not exercised, defer emergency patching and track.
3. **Run exposure mapping.** If the reachable instance has low blast radius, route to the normal patch window.
4. **Check active exploitation.** If exploitation is observed, move to the emergency patch path.
5. **Validate defense coverage.** If compensating controls block the exploit path, buy time and schedule the patch. If a control gap exists, patch or compensate immediately.

This model does not eliminate patching. It scopes and sequences patching. The doctrine creates four possible dispositions:

Disposition	When to Use
Emergency patch	Reachable, exposed, exploited, or insufficiently controlled.
Accelerated validated patch	Reachable and exposed, but no active exploitation and partial control coverage.
Standard patch window	Reachable but low exposure or strong compensating controls.
Defer with evidence	Not reachable or not materially exposed, with documented rationale.

Example: Why the Doctrine Matters

Consider a critical vulnerability affecting nginx across an enterprise fleet of 1,200 instances. A severity-only model may imply that all 1,200 instances require emergency patching.

A runtime decision model may find:

- 720 instances have the relevant module compiled.
- 96 instances have the relevant method enabled.
- 22 instances have the vulnerable configuration path.
- 3 instances are public, reachable, and lack a WAF.

The result is not “do not patch.” The result is:

- Emergency action for 3 maximum-exposure instances.
- Accelerated or standard patch lanes for the other reachable instances.
- Documented deferral for instances where the vulnerable code path is not exercised.
- Detection and compensating controls deployed during validation.

That is the operational difference between panic and prioritization.

Expanded Worked Example: Critical nginx Vulnerability

The original technical report walks through a publicly revealed Mythos disclosure affecting nginx WebDAV. The details matter because they show how quickly a severity-only response can over-scope the emergency lane.

The vulnerability requires a specific combination of conditions:

- The relevant nginx module is compiled into the binary.
- WebDAV methods such as COPY or MOVE are enabled.
- A vulnerable configuration path is present.
- The affected service is reachable by an attacker.

A conventional inventory query might begin with all nginx instances. In the illustrative enterprise fleet, that is 1,200 instances. But the Patch-Gap Defense Doctrine asks narrower questions.

Filter	Illustrative Result	Decision Impact
nginx instances in fleet	1,200	Initial population
Relevant module compiled	720	Dependency is present but not necessarily exploitable
Relevant methods enabled	96	Feature path exists
Vulnerable configuration present	22	Exploit path is reachable
Public, unauthenticated, no WAF	3	Emergency action population

This does not mean the other instances are ignored. It means they are routed differently:

- The 3 maximum-exposure instances require immediate mitigation, detection, and patch planning.
- The remaining reachable instances can move through accelerated or standard validation lanes depending on exposure.
- The non-reachable instances can be deferred from emergency patching with documented evidence.

For the 3 highest-risk instances, the team can deploy compensating controls while the upstream patch is validated:

Control Area	Example Action
Configuration	Disable unsafe WebDAV and alias combinations.
WAF	Block suspicious COPY or MOVE patterns against affected paths.
Runtime monitoring	Alert on unexpected nginx file writes outside approved directories.
Network policy	Restrict egress from the nginx workload.
EDR/SIEM	Watch for post-exploit file creation and process behavior.

The CISO can now make a defensible decision: accept regression risk for the 3 highest-risk systems, stage the rest through a validated lane, and document why 1,178 instances do not require emergency action for that specific exploit path.

That is the business value of runtime evidence. It reduces risk without forcing every critical advisory into a production-breaking patch event.

CISO Quarterly Checklist

- ☑ Build a runtime inventory of the projects and packages implicated by current AI-driven disclosure programs.
- ☑ Tag running instances by dependency presence, code reachability, network reachability, and identity reachability.
- ☑ Map exposure by public/private status, data sensitivity, tenant impact, and business criticality.
- ☑ Pre-deploy detection logic for reachable and exposed instances before public exploitation signals mature.
- ☑ Validate compensating controls against exploit-specific patterns, not generic vulnerability classes.
- ☑ Express control coverage in board- and auditor-defensible language, including MITRE D3FEND or ATT&CK where useful.
- ☑ Track the Anthropic CVD dashboard and transparency log as a high-signal pre-CVE source.
- ☑ Recalibrate severity when AI-generated severity and maintainer severity diverge.
- ☑ Create an internal pending-disclosure board that maps pre-CVE signals to runtime assets.
- ☑ Tabletop the next disclosure wave against a critical dependency such as nginx, Temporal, Nomad, ImageMagick, or wolfSSL.
- ☑ Define the evidence required to move a finding into emergency patch, accelerated patch, standard patch, or defer-with-evidence lanes.

Methodology and Sources

This report analyzes public data from Anthropic's Claude Mythos Preview coordinated vulnerability disclosure program.

Primary sources:

- Anthropic CVD dashboard: https://red.anthropic.com/2026/cvd/
- Anthropic CVD payload.json: https://red.anthropic.com/2026/cvd/data/payload.json
- Anthropic CVD ledger.json: https://red.anthropic.com/2026/cvd/data/ledger.json

Snapshot timestamp:

- 2026-05-22T17:27:03Z

Snapshot fingerprints cited in the source report:

- SHA-256(payload.json): 783ab38b8cd39ab274b9d4e197e57132cdd58a13217d71eacb873ba6d79f2598
- SHA-256(ledger.json): 7e7a8baded3e946c91807652539e39f2cfb91af24b3311694985401b909faf78

Additional enrichment sources referenced in the source report:

- OSV.dev
- GitHub Security Advisory REST API
- GitHub repository metadata
- MITRE D3FEND
- MITRE ATT&CK
- CWE

Important caveats:

- All quantitative claims are snapshot-bound. Anthropic publishes incremental updates, and figures should be refreshed before real-time citation.
- The remediation rate is a lower bound because some fixes may be shipped quietly or without Mythos attribution.
- The vulnerability deficit is measured at the ecosystem level. A specific enterprise's addressable deficit depends on its runtime reachability and exposure profile.
- *The report uses no embargoed disclosure data and does not identify projects still under coordinated disclosure.*

Source Integrity

The report intentionally distinguishes between three kinds of claims:

- **Measured claims:** Counts and rates derived from the May 22, 2026 Mythos snapshot.
- **Derived claims:** Ratios and deficit calculations based on measured counts.
- **Operational claims:** Enterprise implications based on how vulnerability-management, patch-validation, advisory-ingestion, and runtime-control processes work in practice.

This distinction matters for credibility. The 1,596 disclosed vulnerabilities, 97 fixed-in-response count, and 63-day window are measured from the cited sources. The 16.5x ratio and 23.8/day net deficit are derived from those measured values. The recommendation to use runtime reachability, exposure, exploitation, and control coverage is an operating-model conclusion drawn from the data.

What This Report Does Not Claim

This report does not claim:

- Every Mythos disclosure is exploitable in every enterprise.
- Every unpatched disclosure is urgent.
- Coordinated vulnerability disclosure is broken.
- CVE, SCA, SAST, CNAPP, or vulnerability-management tools are obsolete.
- Patch deployment should be avoided.

The claim is narrower and stronger: AI-driven discovery is creating a timing and capacity mismatch that conventional advisory-led workflows cannot resolve alone. Enterprises need runtime evidence to determine which issues require emergency action, which can be safely staged, and which can be deferred with documentation.

From Doctrine to Operation

The Patch-Gap Defense Doctrine is vendor-neutral by design. A mature security organization could assemble pieces of it from SCA, SAST, CNAPP, SIEM, EDR, WAF, BAS, threat-intelligence, service-mesh, CMDB, and homegrown graph tooling.

The operational challenge is stitching those pieces together quickly enough to act during the Patch Gap. The decision depends on context that usually lives in separate systems: dependency presence, runtime call paths, identity reachability, network exposure, exploit signals, WAF behavior, EDR coverage, and compensating-control validation. When those inputs are fragmented, the CISO still ends up making a severity-led decision under pressure.

Tuskira's product relevance is the unification of those inputs into a runtime decision loop. Context Graph supplies the reachable asset and dependency view. Kairo reasons over breach paths, exposure, and detection signal. Crystal validates that the deployed defense policy will catch what Kairo described. Federated Detection distributes the resulting IOA and control guidance into the tools the enterprise already runs. Kairo finds it. Crystal proves the organization can stop it.

That is the practical bridge from research finding to security operation: not "patch less," and not "buy another scanner," but convert every disclosure into a documented, per-instance disposition: emergency patch, accelerated validated patch, standard patch, compensate and monitor, or defer with evidence.

Appendix A: Condensed Bug-Class to D3FEND Map

This condensed map restores the standards anchor behind the Patch-Gap Defense Doctrine. It does not claim that a D3FEND mapping proves a control works. It shows which defensive tactic families are most relevant to each verified Mythos bug class, so teams can express control coverage in board-, auditor-, and analyst-defensible language.

Bug Class	D3FEND Emphasis	Defender Motion
Heap-buffer-overflow	Harden: ASLR, NX, CFI, pointer and integer validation	Confirm memory-safety hardening; prioritize reachable native components.
Auth-bypass	Harden and Detect: MFA, certificate auth, session mediation, auth event thresholds	Validate identity gates and alert on abnormal authentication behavior.
Broken-access-control	Isolate and Detect: permissions, authorization thresholds, resource access patterns	Test authorization boundaries and watch for abnormal resource access.
Type-confusion	Harden and Isolate: type validation, pointer validation, process isolation	Confirm runtime hardening and isolate high-risk execution paths.
Denial-of-service	Isolate and Detect: inbound filtering, session volume analysis, exception monitoring	Rate-limit exposed services and detect volume or crash patterns.
Use-after-free	Harden: reference nullification, pointer validation, ASLR, CFI	Prioritize memory-safe builds and reachable native code paths.
Stack-buffer-overflow	Harden: stack canaries, ASLR, NX, exception-handler validation	Confirm exploit mitigations before routing to emergency patch.
Information disclosure	Harden and Isolate: encryption, credential scrubbing, outbound filtering	Limit data exposure and monitor unexpected transfer patterns.
Privilege escalation	Harden and Isolate: system permissions, user permissions, syscall filtering	Validate least privilege and constrain post-exploit movement.
XSS	Isolate and Harden: content filtering, content conversion, app hardening	Confirm WAF/content controls and session mediation.
SSRF	Isolate and Detect: outbound filtering, DNS allowlists, URL analysis	Restrict egress and detect unexpected internal service access.
Path traversal	Harden, Isolate, Detect: domain logic validation, file permissions, file access analysis	Validate path handling and monitor abnormal file reads/writes.
SQL injection	Detect, Harden, Isolate: query analysis, domain validation, content validation	Validate input handling and monitor abnormal query patterns.
Integer overflow	Harden: integer range validation, pointer validation, CFI, software update	Confirm bounds-checking and memory-hardening coverage.

Operational reading: Harden appears across all 14 verified bug classes and is the highest-leverage control family for preparing the disclosure cohort. Isolate applies to most network-reachable classes, especially auth-bypass, SSRF, broken access control, denial-of-service, and information disclosure. Detect is most useful where exploitation creates observable behavior, such as abnormal authentication, resource access, query, traffic, or file-access patterns.

Appendix B: How Tuskira Supports Patch-Gap Defense

This appendix maps the vendor-neutral operating model to Tuskira capabilities. The research argument above stands on its own; this section explains how Tuskira operationalizes the model inside existing enterprise security stacks.

Recommended Capability	Tuskira Capability	What It Does
Runtime reachability evidence	Context Graph	Builds a runtime call-graph and dependency-presence graph by joining SBOM, eBPF execution telemetry, and IAST-style request tracing into a queryable layer.
Runtime exposure mapping	**Kairo** exposure validation	Walks the breach path against the live Context Graph and validates identity reachability, network reachability, authorization transitions, and blast radius.
Active-exploitation detection	**Kairo** detection-signal emission and Federated Detection	Emits IOA and detection leads such as YARA, Sigma, eBPF policy, and WAF rules, then distributes them across the existing sensor estate.
Compensating-control validation	**Crystal** defense-policy validation	Validates deployed defense policies against the exploit pattern: WAF behavior, network policy, syscall filtering, EDR coverage, and related controls.

Kairo and Crystal operate as a paired agent loop. Kairo identifies the exploitable path and emits detection signal. Crystal validates that the deployed defense policy will catch what Kairo described. Kairo finds it. Crystal proves the organization can stop it. Together, they help security teams make per-disclosure, per-instance decisions throughout the Patch Gap window.

About Tuskira Research

Tuskira Research is the research arm of Tuskira, Inc., an agentic SecOps platform vendor. Tuskira Research publishes analysis of changes in the open-source and enterprise security ecosystem with the objective of contributing to the practitioner, analyst, and journalist community's shared situational awareness.

Correspondence: research@tuskira.ai

Appendix C: Analyst Glossary

Term	Plain-English Meaning
Mythos	Anthropic's AI-driven vulnerability discovery preview program analyzed in this report.
CVD	Coordinated Vulnerability Disclosure: private maintainer notification before public advisory release.
Patch Gap	The window between upstream/private disclosure and safe, deployed enterprise remediation.
Vulnerability deficit	The mismatch between discovery velocity and visible remediation velocity.
CVE	Common Vulnerabilities and Exposures identifier for a publicly known vulnerability.
GHSA	GitHub Security Advisory identifier used in the GitHub advisory ecosystem.
NVD	National Vulnerability Database, a public U.S. vulnerability database used by many tools.
SCA	Software Composition Analysis: dependency and open-source package vulnerability analysis.
IOA	Indicator of Attack: behavior that suggests exploitation is being attempted or is underway.
D3FEND	MITRE's defensive-technique knowledge graph for describing security controls and countermeasures.
EPSS	Exploit Prediction Scoring System: probability that a vulnerability will be exploited in the wild.
VEX	Vulnerability Exploitability eXchange: a machine-readable statement about whether a product is affected or exploitable.