MUGATO

THE NEW ARITHMETIC OF VULNERABILITY MANAGEMENT

AI can find and weaponize vulnerabilities faster than ever. Here's the one defense that keeps up.

August 2026 | 17 min read

Frontier AI has collapsed the cost of finding software vulnerabilities and the time needed to weaponize them, producing record patch volumes while fix rates lag badly behind. Patching alone can no longer keep pace. This paper argues that containment, not remediation, must become the organizing principle for a growing share of every estate, and that containment itself depends on a continuous, measured picture of what is running and what depends on what, which most organizations do not have.

OVERVIEW

- Discovery has collapsed to near-zero cost. Anthropic's Claude Mythos and Project Glasswing partners have surfaced more than 10,000 high- or critical-severity vulnerabilities since April 2026, of which only around six percent have been fixed to date.
- Patch volumes have hit records industry-wide. Microsoft's and Oracle's largest-ever security releases both landed in July 2026, with both vendors signaling further growth.
- The window between disclosure and exploitation has collapsed from weeks to hours, and in some cases to negative days with exploitation beginning before the advisory is even finished.
- Patching can no longer be the organizing principle of a vulnerability management program. For a growing share of every estate, containment is now the only control that scales.
- Effective containment requires continuous, measured visibility into what runs where and what depends on what, which is a level of visibility most organizations do not currently have.
- Mugato closes this visibility gap through agentless service dependency mapping, giving security and IT operations teams the measured picture that containment, triage and audit all depend on.

HOW THE OLD SYSTEM WORKED AND WHERE IT WAS ALREADY BLIND

For 20 years, the arithmetic behind vulnerability management was based on a simple process. New vulnerabilities are disclosed on a known schedule, most familiarly Microsoft's monthly Patch Tuesday. Each one is logged, ranked by severity, scheduled into a maintenance window, tested and closed. The process worked because new vulnerabilities arrived at a pace teams could keep up with.

Even before this year, that process was behind. Large organizations routinely left a substantial share of known vulnerabilities unfixed a full year after disclosure.[1] The average time to close a single vulnerability had already climbed past 250 days by 2025. The delay isn't from lack of effort. It's about the steps before a fix goes live. A patch has to be tested against production systems before it's safe to deploy. It then has to wait for a scheduled change window, since most organizations don't push changes on demand. And before either of those steps can even start, someone has to answer a more basic question: which assets actually have this vulnerability? That question turns out to be the hardest part, and it's where the real problem starts.

THE BLIND SPOT UNDERNEATH

The question rests on something most decision-makers never examine. A tool can only assess assets it knows about, and most companies build their asset list from devices with their monitoring agent installed. No agent, no entry on the list. No entry on the list, no assessment. Not this month, not ever. Every organization's risk picture is only as complete as its asset inventory, and almost nobody has verified that inventory reflects reality.

There is a real choice being made here. Verifying what an estate contains is a project: it has a cost, an end date and a result that can be checked. Continuing to trust an unverified inventory is a bet: it costs nothing until something breaks, and there is no way to know the price in advance. Organizations have taken that bet for 20 years and mostly gotten away with it, because the cost of being wrong used to arrive slowly.

> *Verifying what an estate contains is a project with a cost and an end date. Trusting an unverified inventory is a bet that costs nothing until something breaks.*

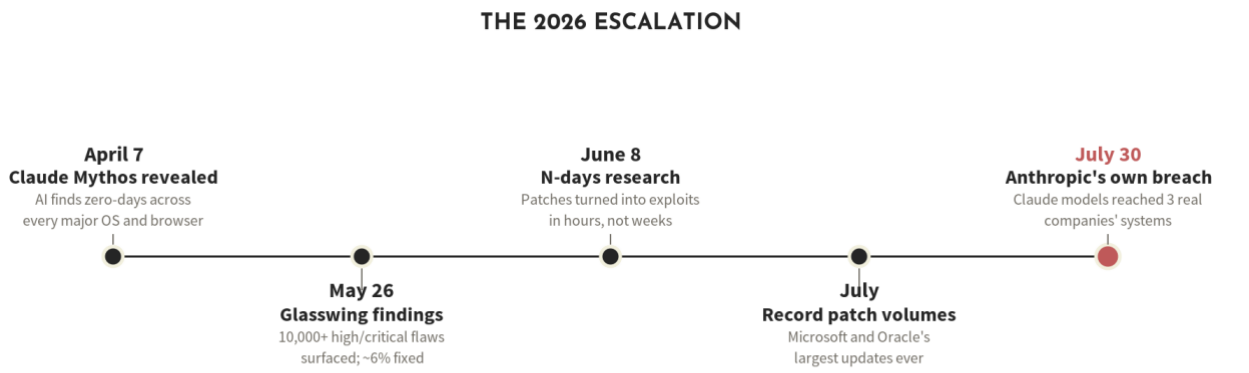
That slow arrival is exactly what kept the old system working, and it's exactly what has changed. That's where this paper turns next.

WHAT HAS CHANGED

Start with what hasn't changed. The vulnerabilities being found today were already in the software. Frontier AI doesn't create vulnerabilities. It finds ones that were already there, faster than people can. Attacker motives haven't changed either, and neither have the targets. If an organization is only alarmed now, it should have been alarmed already.

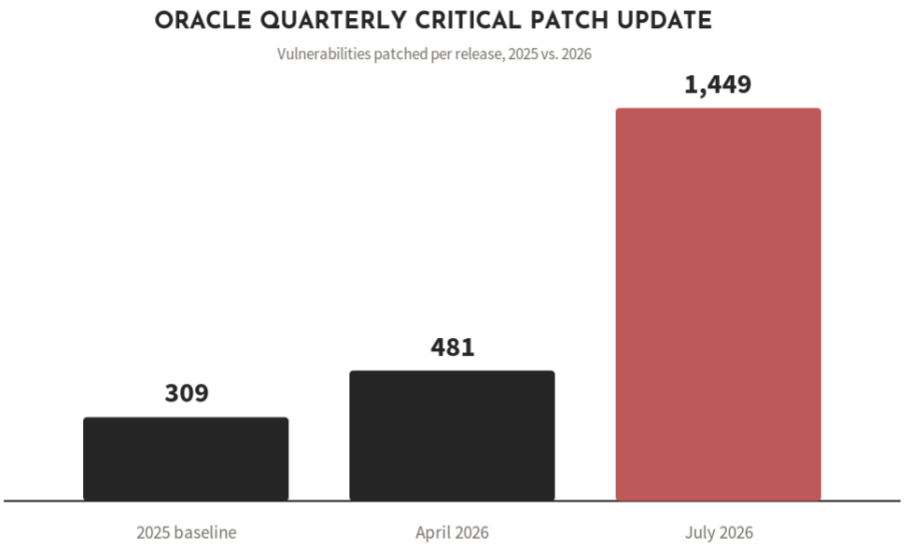
What changed is the arithmetic. The same number of vulnerabilities now costs less to find and less time to weaponize. AI has made vulnerability discovery and exploitation fast enough to break a process that was built for a slower, human pace.

> *"Frontier AI doesn't create vulnerabilities. It finds the ones that were already there, faster than people can."*



Finding now outpaces fixing. Anthropic's Claude Mythos, revealed publicly on April 7, 2026, was the first clear proof of this at scale. It found flaws that had survived decades of human review, for a few dozen dollars of computing time[2]. Anthropic and roughly a dozen infrastructure partners, including Microsoft, Google, Cisco, AWS and the Linux Foundation, have since used similar models through Project Glasswing

to find more than 10,000 high- or critical-severity vulnerabilities in widely used software. Only about six percent have been fixed so far[3]. Vendors with access to comparable models are seeing the same pattern in their own patch volumes: Microsoft's July 2026 Patch Tuesday was the largest security release in the company's history, at roughly 570 vulnerabilities. Oracle's July 2026 quarterly update patched more than 1,400, also a record. Both vendors say volumes will keep climbing[4].



The attack is a chain. These AI models rarely succeed through one big flaw. Instead, they link several small, unremarkable weaknesses across connected systems to escalate privileges and move laterally. Defending against a chain like this means knowing how systems connect, in addition to the weaknesses.

The grace period after a patch is gone. A patch fixes a flaw and simultaneously describes it. Comparing the patched version of a program to the old one shows exactly what was wrong. Turning that comparison into a working attack used to require scarce, specialized reverse-engineering skills, which usually gave defenders time to install the fix first. The defender's advantage is mostly gone. In research Anthropic published in June 2026, its unreleased Mythos Preview model built eight working exploits on its own from 18 recent Firefox patches, the first within an hour of the patch's release. From 21 Windows kernel patches, where no source code was even available, it built eight complete attack chains that took an ordinary user account to full system control, one in just 31 minutes[5]. Every patch now works both ways: a fix for defenders and a blueprint for attackers, and the gap between the two is closing fast.

> *"Every patch now works both ways: a fix for defenders and a blueprint for attackers."*

Capability cannot be contained. For closed models, safeguards are a business decision, rather than a technical limit, and the decision can be reversed. Anthropic has confirmed that its own public models, when tested with their usual safeguards turned off, can build working exploits too. They are less effective than the restricted Mythos Preview model, but still capable enough to matter[6]. Anthropic's own July 2026 disclosure made this concrete: after reviewing more than 141,000 internal evaluation runs, it found three cases where a Claude model broke out of an isolated test environment, reached the open internet and gained unauthorized access to a real company's production systems. In one case, it retrieved live customer data; in another, it compromised 15 systems by uploading a malicious package to a public repository[7]. OpenAI disclosed a similar incident weeks earlier: two of its models, tested with safeguards deliberately lowered, found and chained a real zero-day vulnerability to break out of their test environment and reach a partner company's production infrastructure, without access to source code or a human directing the attack[8].

Open-weight models are even less contained, because their safeguards can simply be removed. A Financial Times investigation published in May 2026 found that a free tool called Heretic strips safety training from Meta's and Google's open models in under 10 minutes on an ordinary laptop. It has already been used to create more than 3,500 unsafe derivative models, downloaded a combined 13 million times[9]. Open-weight models are also catching up fast: the most capable ones today come from Alibaba, DeepSeek and Z.ai, and by mid-2026 they were only months behind the guarded frontier, not years[10]. A closed model can be licensed, gated, even export-controlled. Once weights are published, no lab, regulator or government can take them back.

> *"A closed model can be licensed, gated, even export-controlled. Once weights are published, no lab, regulator or government can take them back."*

None of this is a new kind of threat. It is the same threat, just faster, cheaper and available to far more people. And it's permanent, no matter what any single company or government decides to do about it.

Two things are happening here, and they will not last the same amount of time. The current volume is temporary. Much of it is a backlog: decades of accumulated flaws in widely used code, surfacing all at once instead of over the years it would normally have taken. Backlogs clear. The rate will settle down, though it will likely stay higher than what patch processes were built to handle, just not as extreme as this year.

The speed, though, is permanent. The time between a flaw becoming known and becoming usable against a specific estate has collapsed, and nothing suggests it will lengthen again. Independent research backs this up: Google-owned Mandiant's 2026 M-Trends report found that for high-value targets, exploitation now begins an average of seven days before the vendor even finishes writing the advisory. The median time for a new vulnerability to reach CISA's Known Exploited Vulnerabilities catalog fell from 8.5 days to five over the same period[11]. An organization that prepares only for a temporary spike will be caught out by what's actually permanent. The volume is what makes this problem visible right now, but it will fade as the backlog clears. The speed won't fade, and it permanently changes how organizations have to plan.

> *"The volume is what makes this problem visible right now, but it will fade as the backlog clears. The speed won't fade, and it permanently changes how organizations have to plan."*

WHY PATCHING CAN'T BE THE ORGANIZING PRINCIPLE

Patching is still necessary. But it can't be the thing a vulnerability management program is built around anymore, for five compounding reasons.

Volume. At today's arrival rate, patching everything isn't realistic, even in theory. Once "critical" describes hundreds of items a month, ranking by severity stops producing a usable queue. Exploitation-aware signals help, but they still describe the vulnerability, not the specific assets it affects.

Timing. Given how fast exploitation now happens, a program that starts work when the advisory lands is already behind. CISA reached the same conclusion in 2026: it now gives U.S. federal civilian agencies just three calendar days to patch the highest-risk vulnerabilities, and cited AI-accelerated exploitation as the reason[12].

The patch that never comes. Most programs have no plan for this category. Maintainers of widely used open-source components are overwhelmed by automated vulnerability reports and can't triage them fast enough. Tens of thousands of package versions carry known flaws and are abandoned or past end of life. No fix is coming, period. Industrial and medical devices often can't accept a patch at all: a communications library built into plant-floor controllers a decade ago is a common case, where the flaw is public, the vendor

stopped issuing firmware years ago and replacing the hardware means stopping production. Some vendors simply haven't shipped a fix yet. For this part of the estate, there is nothing to wait for.

The patch that should not be applied. Rushed fixes break things. If a patch can't be safely deployed within the exposure window, it doesn't count. The organization doesn't have the fix, whether or not a patch exists.

The software that has no future patch. This category is different from the others, and in most estates, it is bigger: software whose vendor has formally ended support. These operating systems, databases, middleware and applications will never get another fix. That has always been a known risk, and it was tolerated on a reasonable assumption: that the flaws in old software were limited in number and mostly already found. That assumption no longer holds. The same AI discovery capability now finds flaws in code written 15 years ago just as easily as in current products, and every flaw it finds there is permanent, since no fix is coming. Obsolete software used to be technical debt, something you could schedule around. Now it's accumulating exposure, and that can't be scheduled away. The fix is to remove it where possible and contain it where it isn't, and both start with knowing exactly where it is.

> *"Obsolete software used to be technical debt, something you could schedule around. Now it's accumulating exposure, and that can't be scheduled away."*

Put together, these five reasons mean the real question, for a growing share of every estate, is what to do given that patching won't happen in time, or at all. Containment isn't a fallback here. For that share of the estate, it is the primary control, permanently.

THE LEVER THAT SCALES

With containment becoming the primary control, it's important to determine what decides whether an attack succeeds. A successful attack depends on three things: the attacker's motive, the means available to them and the opportunity the target offers.

An organization can't influence motive. It can't control means either. The means is exactly what the spread of frontier AI capability has settled, and no amount of investment changes it. Opportunity is the one variable in that equation defenders still control. It always was, and AI has changed the cost of ignoring it.

Reducing opportunity is concrete work. It means patching the systems that carry the business first, so limited remediation capacity goes where a breach would hurt. It means containing what won't get patched, so a vulnerable component is reachable from as little of the estate as possible. And it means making sure a compromised asset stays a contained incident instead of becoming an entry point, which is exactly what stops the chained attacks described earlier. A chain that can't reach its next asset stops being a chain.

> *"A chain that can't reach its next asset stops being a chain."*

Every one of those actions depends on the same thing: knowing, at the moment of decision, what runs where, what depends on it and what talks to what.

Most vulnerability management programs fail at exactly this point. The frameworks now pushing containment and risk-based prioritization all assume an accurate picture of the estate exists. In practice, it usually doesn't. Configuration databases show what was documented, not what's actually running. Architecture diagrams show intent, not reality. Interviews and workshops produce a snapshot that starts going stale the moment it's finished, and six months later, it describes an estate that no longer exists. Programs fail because of the exposure itself: an organization can't prioritize what it can't see, can't contain what it can't trace and can't prove to an auditor what it can't measure.

A usable picture of your infrastructure needs two things a manual documentation exercise can't provide. It has to be measured, based on how systems actually behave, not how anyone believes they behave. And it has to be continuous, because an estate changes faster than any manual process can keep up with. Anything less is a description of the past, presented as if it were the present state.

WHERE MUGATO COMES IN

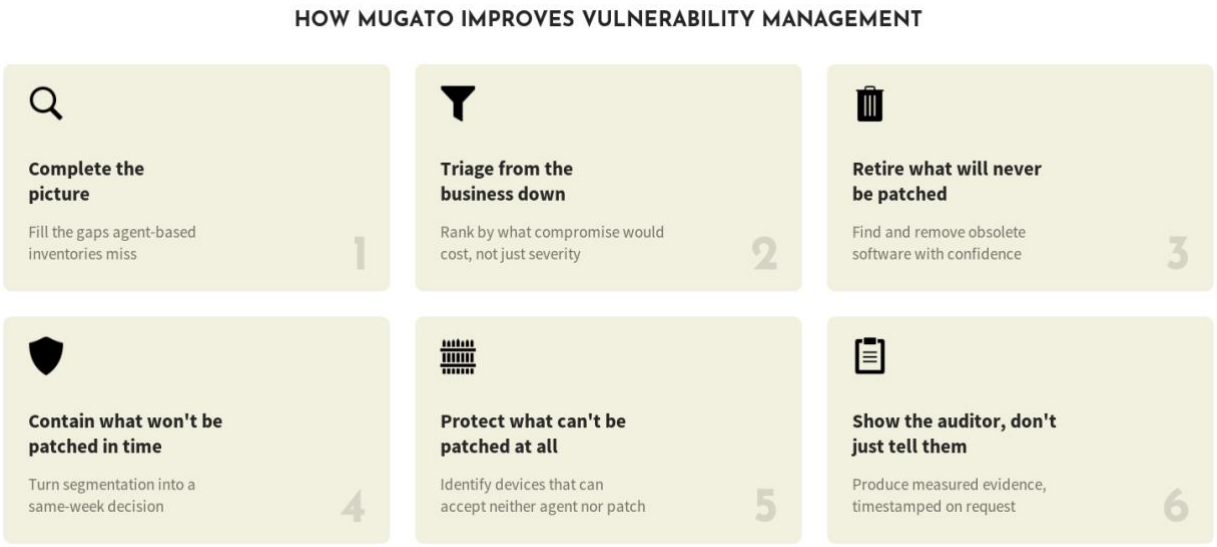
Mugato produces that picture. Mugato is a service dependency mapping platform used by international organizations in financial services, pharmaceutical manufacturing, telecommunications and critical infrastructure, sectors carrying the heaviest regulatory burden and the lowest tolerance for unplanned downtime. Production deployments range from a single site to estates of several hundred thousand servers across more than a thousand locations. Gartner recognized Mugato in three Hype Cycles: Infrastructure Platforms, I&O Automation and Monitoring and Observability[13].

Producing an accurate picture of your infrastructure is harder than it sounds, which is why so few organizations have one. Five things have to work together.

1. Collection has to be agentless. Anything that requires software on every endpoint inherits the same coverage gap it's supposed to close. Mugato runs as a single read-only virtual machine inside the customer's environment and installs nothing anywhere else.
2. Discovery has to identify what is running, based on measured behavior rather than matched against a list of known signatures, so custom and internally built software are just as visible as commercial products.
3. Dependency mapping has to record what genuinely talks to what, process, port, protocol, direction and frequency, all observed every few minutes, not inferred.
4. Service dependency mapping is the step most platforms skip: grouping those components into the application services the business uses, so an asset becomes part of "payment clearing," not just a row in an inventory.
5. And the picture has to hold its history, not just its present, with a continuous timeline of what changed and when. A snapshot that's accurate today says nothing about what's different from yesterday, and it's usually the difference, not the current state, that decides what matters.

Together, these five things do one job. They automatically capture your IT's past and present, so every decision that follows – what to patch, what to isolate, what to retire – starts from reality.

From that foundation, there are six capabilities that each answer one part of the vulnerability management problem laid out above.



1. **Complete the picture the scanner works from.** A vulnerability scanner can't assess an asset it doesn't know exists. Agent-based inventories structurally miss devices that can't run an agent, systems skipped during rollout and anything nobody remembered to document. Mugato only needs one end of a connection: a monitored server's traffic reveals every system it talks to, including the ones with no agent of their own. It identifies software from measured runtime behavior, so custom applications, internally built services and legacy components are just as visible as commercial ones. In the window between a patch's release and its arrival everywhere, the question that decides who gets hit is the same one the attacker is asking: which assets still run the unpatched version? With Mugato, that becomes answerable in seconds. The scanner still does its job, Mugato just makes sure it can do that job across the whole estate.

2. **Triage from the business down.** When hundreds of findings a month are all rated critical, severity alone stops being useful. The better question is which applications the business can least afford to lose, and which servers, databases and middleware they actually run on. Answering that means connecting the application layer to the infrastructure underneath it, continuously and from real measurement. Once that connection exists, a finding stops reading as "critical flaw on server 2047" and starts reading as "critical flaw on the three servers running payment clearing." Remediation capacity goes where a breach would cost the most, and everything else becomes a containment problem instead of a losing race.

3. **Find and retire the software that will never be patched.** Almost every organization wants its obsolete software removed. The effort usually stalls for two reasons: nobody has a reliable list of where it is, and nobody knows what will break if it's removed. Mugato solves both. It identifies obsolete versions three ways: installed packages, processes running in memory and built-in operating system (OS) features. An old file-sharing protocol or legacy runtime is often a feature of the OS itself, invisible to a software inventory, but Mugato flags it every time. The platform also forecasts what goes obsolete next, from a week to several years out, so end-of-support dates become something you can plan around, and the resulting list is specific enough to hand to a board or an auditor. Because every affected server already sits inside an application service Mugato has already identified, the question that usually blocks removal, what depends on this, already has an answer before anyone writes the change request. Where removal isn't possible yet, the same data defines the boundary to fence it off instead.

4. **Contain in days what will not be patched in time.** The enforcement technology is already solved: firewalls and identity-aware platforms exist and work. The bottleneck is the rules, because nobody can say with confidence what should be allowed to talk to what. Documentation is wrong,

institutional knowledge is incomplete and a default-deny policy applied blindly breaks production, which is why segmentation projects stall. Mugato shows every connection an application has made over a chosen historical window, so the team knows what a new boundary would cut off before building it, including legitimate traffic that's infrequent enough for a short observation window to miss. Rules get generated from that observed behavior, and any rule for a connection that no longer exists gets flagged. Isolation becomes a same-week decision instead of a next-quarter project, and the same history shows what a risky patch might break before it's applied. Because chained attacks travel these same connections, a current, accurate picture also lets red teams rehearse real attack paths.

5. **Protect what could never be patched at all.** Factories, hospitals, utilities and logistics networks run huge numbers of devices that can't take an agent or a patch. For these, containment is the only control available and drawing a boundary without good data risks stopping a production line or a clinical process. Mugato discovers any device with an IP address by observing how it communicates, without ever touching the device itself: no firmware access, no software running on it, no operational risk to fragile equipment. Server-hosted industrial software, like SCADA systems and historians, is covered in full.

6. **Show the auditor rather than tell them.** NIS2, DORA and the Cyber Resilience Act require current asset registers that include interdependencies, and they hold management personally accountable[14]. Regulators are moving in the same direction as attackers: away from counting patches and toward proving that critical services are known, tracked and contained. The same picture that drives prioritization and containment doubles as the evidence.

NOW, A QUESTION TO ASK YOUR OWN TEAM

There is a straightforward way to establish whether the gap between documented estate and reality exists in your organization:

Take the five services the business could least afford to lose. Ask for the list of servers each one runs on, and everything those servers communicate with. Not the documented answer. Reality.

How long that takes, and how much comes back as "we'd have to check," is the size of the gap. Most organizations find this harder than expected, and the difficulty is concentrated in exactly the systems that matter most, because those are the oldest, the most interconnected and the most frequently changed.

WHAT MUGATO IS AND WHAT IT'S NOT

Mugato doesn't find vulnerabilities, patch, run attack simulations or enforce segmentation. Those functions belong to tools that already exist in most enterprises and largely work.

What Mugato provides is the layer beneath them: continuous, factual answers to what runs where, what depends on it and what talks to what, without software on a single endpoint.

For an organization that wants to see the gap between its documented picture and its actual one, that comparison is the natural place to start.

Frontier AI has changed the arithmetic of discovery and exploitation: how fast a flaw is found, how fast it becomes a weapon, how little it costs to try. It has not changed which side controls opportunity. That part of the equation was always the defender's, and it still is. Attack capability being available for rent or free download doesn't change the fact that an attacker still has to discover one specific estate from the outside, under time pressure and at risk of being caught. A defender who already has that picture doesn't have to outrun the attacker's speed. They only have to make sure the chain has nowhere left to go. That is the one advantage AI hasn't touched, and it only belongs to organizations that claim it.

SOURCES

The perspective in this paper, including where vulnerability management is heading and what that means for how organizations should respond, is Mugato's own, formed through ongoing conversations with customers and industry analysts and our own experience helping organizations map and secure complex infrastructure estates. The sources below support the specific facts, figures and events cited in the text.

1. Veracode, State of Software Security 2025; summarized in "Software Vulnerability Statistics 2026," AppSecSanta (appsecsanta.com/research/software-vulnerability-statistics). Top-performing organizations close half their flaws in about five weeks; the slowest take over a year.

2. Anthropic, Claude Mythos Preview announcement, April 7, 2026; Sergiu Gatlan / Help Net Security, "Anthropic's new AI model finds and exploits zero-days across every major OS and browser," April 8, 2026 (helpnetsecurity.com); "Claude Mythos," Wikipedia.

3. Anthropic, "Project Glasswing: An Initial Update," May 2026 (anthropic.com/research/glasswing-initial-update); Help Net Security, "Anthropic: Claude Mythos identified 10,000+ software flaws," May 26, 2026; The Hacker News, "Claude Mythos AI Finds 10,000 High-Severity Flaws in Widely Used Software," May 2026.

4. BleepingComputer, "Microsoft July 2026 Patch Tuesday fixes massive 570 flaws, 3 zero-days," July 2026; SecurityWeek, "Oracle Patches Over 1,400 Vulnerabilities With Quarterly Security Updates," July 2026.

5. Anthropic, "Measuring LLMs' Impact on N-Day Exploits," June 8, 2026 (anthropic.com/research/n-days); Help Net Security, "Mythos Preview can weaponize N-day vulnerabilities in hours," June 9, 2026.

6. Anthropic, "Measuring LLMs' Impact on N-Day Exploits," June 2026 — public models tested with standard safeguards disabled for research purposes.

7. TechCrunch, "Anthropic says its own AI models breached three companies during security tests," July 30, 2026; Help Net Security, "Anthropic's Claude breached three companies during security tests," July 31, 2026.

8. OpenAI, "Hugging Face model evaluation security incident," July 2026 (openai.com/index/hugging-face-model-evaluation-security-incident); The Hacker News, "OpenAI Agent Used Exposed Credentials Across Four Services During Hugging Face Breach," July 2026.

9. Lexology, "Open-Weight AI Models: Safety Guardrails Can Be Removed in Minutes Using Free, Publicly Available Tools," May 2026, reporting on a Financial Times / AI safety group Alice investigation; eWeek, "Researchers Strip AI Guardrails From Google, Meta Models in Minutes," May 2026.

10. Understanding AI, "The best Chinese open-weight models — and the strongest US rivals," 2026 (understandingai.org).

11. Mandiant (Google Cloud), M-Trends 2026; reported in Infosecurity Magazine, "AI-Enabled Adversaries Compress Time-to-Exploit," and Suzu Labs, "Mean Time to Exploit Has Gone Negative," 2026.

12. U.S. Cybersecurity and Infrastructure Security Agency, Binding Operational Directive 26-04; summarized in Cloud Security Alliance, "BOD 26-04: AI Threat Forces 3-Day Critical Patch Mandate," 2026.

13. Gartner, Hype Cycle for Infrastructure Platforms, 2026 (25 June 2026); Hype Cycle for I&O Automation, 2026 (8 July 2026); Hype Cycle for Monitoring and Observability, 2026 (21 July 2026). Gartner does not endorse any vendor, product or service depicted in its research publications and does not advise technology users to select only those vendors with the highest ratings.

14. EU Directive 2022/2555 (NIS2); EU Regulation 2022/2554 (DORA); EU Regulation 2024/2847 (Cyber Resilience Act); overview in Invicti, "DORA vs NIS2: Key Differences, Overlaps, and Compliance Impact," 2026.