Virtual Patching for the Frontier AI Era

The Threat Has Changed

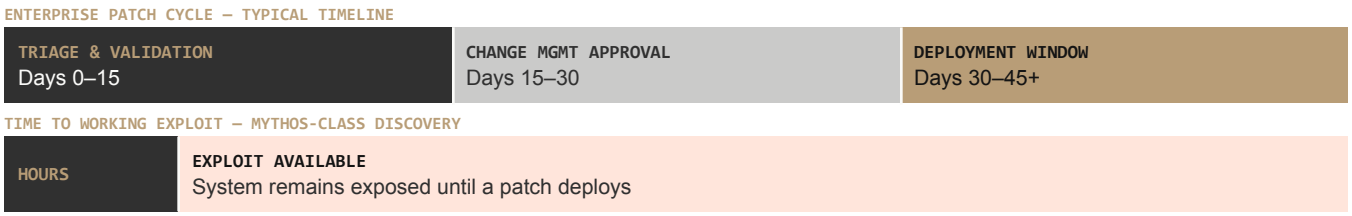
Frontier AI models have compressed the interval between vulnerability discovery and a working exploit from weeks to hours. That compression lands on a timing problem that already existed: *even before AI-accelerated discovery, the average time from disclosure to a working exploit had fallen to roughly five days, while a typical enterprise patch cycle for the same vulnerability runs 30 to 45 days or more (see the cadence comparison below*). Patch-dependent protection assumes that the gap is survivable. It was already tight. Mythos-class discovery makes it effectively zero: *a working exploit can exist before triage even starts.*

Agentic AI compounds this from inside the perimeter. It operates with valid credentials and approved access, generating change at a velocity human-in-the-loop review was never built to handle. The gap between what CISOs are asked to enable and what they can see, control, or audit widens every quarter.

Why Current Approaches Fall Short

The standard remediation cycle, detect, analyze, fix, assumes humans are in the loop and attackers move at human speed. AI agents break both assumptions. Detection-based controls need prior knowledge of an attack pattern, but Mythos-class discovery generates new patterns faster than analysts can write rules for them. Patching, the other primary remediation path, is slow by design and often impossible outright: *end-of-life systems and third-party software the enterprise can't modify may never receive a fix at all.* Either way, the exposure window between a vulnerability becoming known and a fix deploying is now a window the attacker moves through before the security team finishes triage.

Patching Cadence Math



Even before AI-accelerated discovery, the average time from disclosure to a working exploit had fallen to roughly five days (industry research, 2025), shorter than the fastest stage of a typical 30- to 45-day enterprise patch cycle (Verizon DBIR 2024; CISA BOD 19-02). Mythos-class discovery compresses that exploit timeline to hours without changing the patch cycle, so the entire cycle becomes the exposure window.

Mimic's Response: Virtual Patching

Architecture. Mimic enforces a known-good policy for each protected component at the kernel level. The baseline is established by profiling the system in its approved operational state during onboarding, then maintained as the environment evolves. From that point, every change to a protected component, a new binary, a modified registry key, a tampered driver, an unapproved configuration, is evaluated at the kernel before it executes. Authorized changes proceed normally. Anything outside the baseline does not. The distinction is the question being asked: *detection-based controls ask whether activity matches a known-bad pattern. Mimic asks whether the change was authorized.* The first question requires knowing the attack in advance. The second doesn't.

Real-time application identification. Mimic maintains a continuously updated inventory of every application in the environment and how they relate, captured at the kernel where process, file, and configuration attribution is unambiguous. When a change violates the baseline, the affected application is identified at the moment the change is attempted, not after log correlation, analyst triage, or lateral movement. Identification and enforcement happen in the same kernel call.

Containment via Mimic's protection boundaries. Mimic's protection boundaries operate at the kernel and define what each application can do: *which files it can touch, which configurations it can modify, which components it can reach.*

Network ACLs gate traffic between hosts after a process is already compromised. Mimic's boundaries gate what a compromised process can do inside the host, before it acts. When a breach is identified, the boundary already in place holds the application to its known-good scope. Lateral movement and privilege escalation outside that scope aren't available to the attacker.

Closing the Frontier AI Vulnerability Gap

Mythos-class discovery doesn't just surface more vulnerabilities. It reaches places human-paced research rarely did: *end-of-life systems, decades-old protocol code, and third-party dependencies the enterprise doesn't control.* Most of those will never get a vendor patch. Under a patch-dependent model, a Mythos-discovered vulnerability in an unsupported system is permanent exposure: *the fix that would close it isn't coming.*

Virtual Patching changes what "discovered" means for that exposure. Because Mimic enforces known-good at the kernel rather than recognizing the exploit, the path from vulnerability to impact still requires an unauthorized change: *a binary that shouldn't run, a registry key that shouldn't change, a driver that shouldn't load.* Mimic evaluates that change against the baseline regardless of which vulnerability produced it, or whether a patch for it exists. A vulnerability Mythos or similar models finds today is covered the same way as one nobody has found yet.

Where Traditional Controls Fall Short

The blast radius math changes with containment speed. A breach contained at the application level in real time is an isolated incident the business absorbs without operational impact. The same breach, allowed to propagate while detection and network controls catch up, becomes downtime, regulatory exposure, and recovery cost.

WAFs match payloads. IPSs match signatures. EDR exploit-mitigation modules match behaviors an analyst has already studied. Network ACLs see traffic between hosts but can't act on what a process does once it's already inside one. Every one of these depends on prior knowledge of the attack, a model that degrades as frontier AI discovery generates novel exploits faster than they can be studied and written into rules. The patch cycle has the same dependency on a slower clock: vendor patch availability moves at vendor engineering velocity, which hasn't changed, while attacker discovery velocity has moved by orders of magnitude.

This holds as the threat evolves. Every successful exploit still requires an unauthorized change to a protected component to achieve impact. As long as that's true, faster discovery doesn't change what known-good enforcement catches.

What a Skeptical CISO Would Actually Ask

Does Virtual Patching replace our patch management program?
No. Patch management still resolves the underlying vulnerability, that's the long-term fix. Virtual Patching closes the exposure window while that process runs, and keeps holding for the systems patch management can't reach: end-of-life platforms, unsupported third-party components, anything with no patch coming. They run on different timelines toward the same outcome.

When the vendor patch finally ships, do we have to do anything to remove Virtual Patching's coverage?
No. Mimic isn't tracking that CVE or its patch status, it's enforcing the system's known-good state regardless of which vulnerability a given unauthorized change happens to relate to. The vendor patch lands as an authorized change to the baseline, and enforcement continues without modification. There's nothing to unwind.

What protects this control if Mythos can find vulnerabilities in any codebase?
The relevant question is not whether code is theoretically findable. All code is. The question is whether compromising Mimic yields the outcome an attacker requires. The architecture is designed so that a compromise of the agent cannot bypass the enforcement layer beneath it.

Detection can't keep pace with AI-driven discovery. Patching can't keep pace with frontier model compression.

Virtual Patching does.