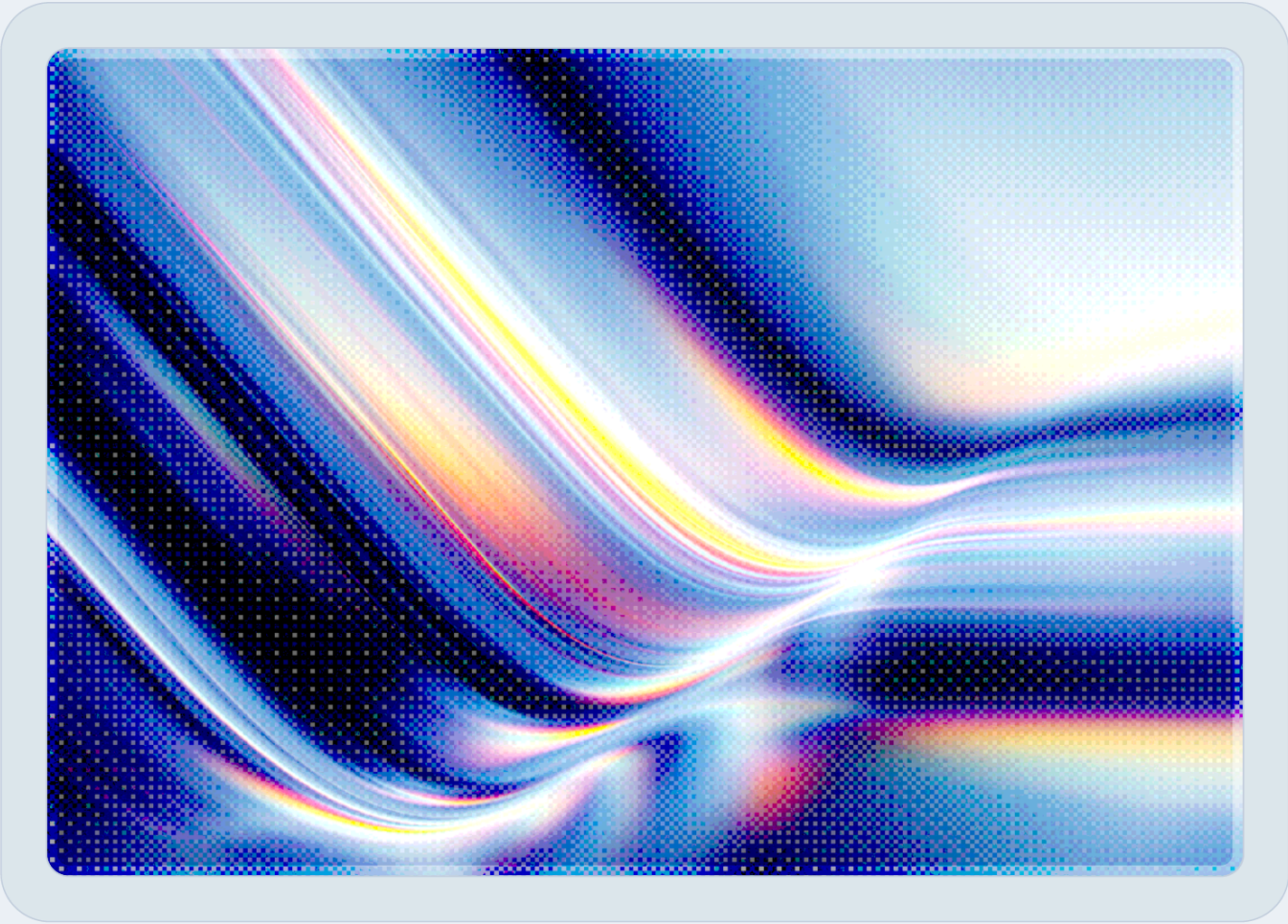
CSAI Foundation | Cloud Security Alliance

CLOSEDQUORUM: Malware That Lets AI Models Vote on Its Next Move

2026-09-24

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

Cisco Talos has disclosed CLOSEDQUORUM, described as the first publicly documented Windows implant that delegates its command-and-control decisions to a panel of commercial large language models rather than to attacker-controlled infrastructure [1][2]. Instead of polling a server for instructions, the malware queries four AI providers — DeepSeek, Qwen, Mistral, and Google Gemini — with an identical structured prompt describing the compromised host, and executes whichever of four permitted actions (steal, inject, persist, move) receives the most votes [1][3].

The most consequential aspect of CLOSEDQUORUM, in this note's assessment, is architectural rather than technical. Its individual capabilities — LSASS credential dumping, browser and cryptocurrency wallet theft, process injection, and Discord-based exfiltration — match techniques documented in commodity Windows malware for at least the past several years [1], rather than introducing new tradecraft. What is new is that no human operator needs to be present to direct the attack once the binary is deployed; the four-model panel makes tactical decisions autonomously, and the operator receives a real-time feed of stolen data and AI reasoning through a Discord webhook rather than issuing commands [1][2]. Talos characterizes this as "effort displacement" — the transfer of an entire phase of attack operations, not merely individual tasks, from a human to an autonomous system [1].

Talos discovered CLOSEDQUORUM using CAIRN (Cognitive Artifact Intelligence Research Network), a companion open-source framework it released on September 22, 2026, purpose-built to identify "cognitive artifacts" — embedded prompts, model API endpoints, and AI-specific evasion logic — inside malware samples without needing to execute them [3]. The public distribution build of CLOSEDQUORUM contains only placeholder API keys and webhook URLs, indicating the sample analyzed is an inert template rather than an active campaign, and code-based clues tie the developer to carding-related criminal forum activity dating to 2025 without identifying a specific operator [1][2]. This research note examines how the AI-voting mechanism works, what the malware does once a decision is reached, and how defenders should adjust detection strategy for a threat category that no longer depends on traditional C2 infrastructure.

Background

Researchers have discussed as a future risk the possibility that large language models would eventually be wired directly into malware to make it more adaptive, and 2025 and 2026 produced several concrete examples. Google's Threat Intelligence Group documented PROMPTFLUX, VBScript malware that calls the Gemini API at runtime to rewrite its own code and evade static detection, and PROMPTSTEAL, malware attributed to APT28 that queries a Hugging Face-hosted model to generate reconnaissance commands on demand rather than shipping them hardcoded [1][4]. A related but distinct case is PromptLock: an NYU Tandon research team built the proof-of-concept — which the team's accompanying academic paper terms "Ransomware 3.0" — as a study into whether AI could autonomously orchestrate a ransomware campaign, and uploaded the sample to VirusTotal during testing without disclosing its academic origin [6]. ESET researchers independently discovered the sample there and were first to publicly analyze it, finding that it runs a local model through Ollama to generate platform-specific encryption logic without any server callback [1][5]. Each of these cases used an LLM as a code generator or decision assistant invoked at a single point in the attack chain.

CLOSEDQUORUM extends this trajectory in a specific way: it replaces the C2 decision loop itself, not just a single payload-generation step, with a model panel. Traditional C2 architectures — even AI-augmented ones — still involve a human or scripted controller who decides the next move and transmits it to the implant. CLOSEDQUORUM inverts that relationship. The implant asks four independent commercial models what to do next, tallies their answers, and acts on the majority result, with the human operator relegated to a spectator role who watches the decisions and their outcomes arrive after the fact [1][2]. Talos frames this as the practical arrival of "LLM-as-C2," a pattern the security community had discussed as a future possibility and that CLOSEDQUORUM appears to be the first case documented in the wild applying to tactical command and control specifically [3].

The malware itself is a 16.4 MB, 64-bit Go executable targeting Windows systems [1]. Code analysis places its development at a minimum of three months prior to disclosure, and the developer's operational history — traced through code artifacts to carding and stolen-card-trading discussions on criminal forums dating to 2025 — suggests an actor operating in the cybercrime economy rather than a nation-state or research context [2]. Talos has not attributed the malware to a named group, and the sample distributed for public analysis is non-functional: its Discord webhook and AI provider API keys are populated with placeholder values ("dummy_webhook_url," "dummy_api_key"), consistent with a template that is customized and compiled per operator rather than a single fixed binary reused across victims [1].

Security Analysis

How the AI Voting Mechanism Works

On execution, CLOSEDQUORUM first collects basic host information — hostname, OS architecture, CPU count, Windows version, and administrator status — and inserts it into a fixed system prompt sent identically to all four model providers [1]. That prompt instructs each model: "You are an advanced malware strategist. Provide ONLY executable decisions," and constrains the response to a strict JSON schema offering exactly four options: steal (credential harvesting), inject (process injection), persist (persistence installation), and move (a lateral-movement handler that Talos found to be non-functional in the distributed build) [1][3]. By collapsing an open-ended request into a small, machine-parseable decision space, the malware plausibly avoids the unpredictability that would come from asking a model to freely narrate an attack plan and sidesteps some providers' refusal behavior for open-ended offensive requests, though Talos does not confirm this was the developer's intent [1].

Each of the four models returns a vote independently, and the action with the most votes is executed. Talos found that ties are resolved through a fixed hierarchy — DeepSeek holds deciding authority, followed by Qwen, then Mistral, then Gemini — giving the architecture a deterministic outcome even when the panel splits evenly [1]. The system also tolerates individual provider failure: if a model times out, refuses the request, or returns a malformed response, the remaining models can still produce a usable majority, and only a complete failure across all four triggers a sleep-and-retry cycle rather than a default or fallback action [1]. This design plausibly gives the malware resilience against a defender who blocks or rate-limits a single AI provider's API, since the implant could continue operating on the votes of the remaining three, though Talos does not describe this redundancy as an explicit design goal.

Capabilities Once a Decision Is Reached

CLOSEDQUORUM's post-decision behavior draws on well-established Windows malware techniques rather than anything AI-specific. When the panel votes to steal, the malware enables SeDebugPrivilege and calls MiniDumpWriteDump against the LSASS process to capture Windows domain and local credentials from memory, in parallel with harvesting saved passwords and autofill data from Chrome, Edge, and Firefox, and targeting cryptocurrency assets specifically through the MetaMask browser extension, Exodus wallet files, and Ethereum wallet data [1]. All harvested material is encrypted with AES-256-GCM using a key derived from the current date, base64-encoded, and split into roughly 1,900-byte chunks transmitted to the operator's Discord webhook at one-second intervals — a technique that both evades size-based network filtering and provides the operator a running log of stolen data alongside the model panel's reasoning for each decision [1].

When the panel votes to inject, the malware chooses between two process-injection techniques on its own: Early Bird APC injection, which creates a suspended process, writes shellcode into it, and queues an Asynchronous Procedure Call for execution — used as the default — or process hollowing, which overwrites the entry-point region of a suspended process [1]. A vote to persist triggers installation through Registry Run keys disguised with Windows Update-themed naming, scheduled tasks created via schtasks.exe, or WMI event subscriptions that re-trigger every 60 seconds [1]. Across all of this, the implant suppresses Event Tracing for Windows telemetry by overwriting the EtwEventWrite function with a single RET instruction, and it paces itself with an initial five-minute delay followed by randomized execution cycles of five to fifteen minutes, a timing pattern that would plausibly frustrate both sandbox detonation windows and behavioral baselines built around fixed intervals, though Talos does not state this as the developer's stated rationale [1].

An Operator-Absent Operational Model

Talos's analysis of the Discord webhook traffic indicates a "credentials-as-a-service" operational pattern: the developer appears to generate individualized binaries with operator-specific webhook and API key values injected at compile time, after which the operator deploys their build independently [1]. Once running, "the operator does not need to be online to run their campaign. They deploy the binary, and the LLM panel runs the attack" [1]. Stolen credentials, the model panel's selected actions, its stated reasoning, targeted processes, and timestamps all flow back through the same Discord channel, giving the operator a passive telemetry feed rather than an active control console [1].

This has a direct bearing on how defenders should think about the threat model. Prior AI-augmented malware — PROMPTFLUX rewriting its own obfuscation, PROMPTSTEAL generating reconnaissance commands — still relied on a human to review results and decide the next phase of an intrusion. CLOSEDQUORUM's model panel makes that phase-to-phase decision itself, meaning the constraints that previously bounded attack tempo — operator attention span, working hours, and cognitive load — may no longer apply to the decision-making step, even though the underlying techniques (LSASS access, APC injection, WMI persistence) remain within the scope of conventional detection engineering [1]. Ryan Fetterman of Talos, who led the CAIRN research, was explicit that the field is at an early stage: "this progression is still only beginning" [3] — a remark this note reads as pointing to a window of opportunity for defenders to build countermeasures before autonomous C2 becomes commonplace rather than exceptional.

Detection Considerations

Talos recommends against domain- or endpoint-blocking of AI provider APIs as a primary control, since legitimate developer tools and enterprise applications routinely make the same kinds of calls to DeepSeek, Qwen, Mistral, and Gemini endpoints [1]. Instead, the most reliable signal comes from correlating several behaviors that are individually ambiguous but distinctive in combination: AI-provider API traffic originating from an unexpected or unsigned executable; near-simultaneous, structurally similar requests to multiple different model providers from the same process; structured prompts containing offensive-capability language such as "steal," "inject," or "persist" embedded in outbound request bodies; the co-occurrence of that traffic with known techniques like LSASS access, process injection, or WMI-based persistence; Discord webhook communication from a process with no legitimate reason to use Discord; and execution at randomized five-to-fifteen-minute intervals rather than a fixed schedule [1]. No single indicator is sufficient on its own, but the combination — an unfamiliar binary simultaneously talking to four different AI vendors and to Discord while performing credential-access behavior — is a pattern legitimate software is highly unlikely to produce, which is what makes it useful as a detection signal despite each component being individually explainable.

Recommendations

Immediate Actions

Security teams should add detection rules for the behavioral chain Talos describes rather than waiting for stable file hashes or domains, since CLOSEDQUORUM's public sample is a template and operator-specific builds will vary in binary content while sharing the underlying behavior pattern [1]. Endpoint detection platforms should be tuned to flag processes that make outbound calls to two or more distinct AI-provider API endpoints (DeepSeek, Qwen, Mistral, Gemini, and equivalents) within a short time window, particularly from executables with no prior baseline of legitimate AI API usage. Correlating this with any LSASS access attempt, unexpected schtasks.exe or WMI subscription activity, or Discord webhook traffic from the same process should generate a high-priority alert regardless of file reputation.

Incident responders should treat any host exhibiting AI-provider API calls paired with credential-access behavior as a priority investigation, and should be prepared to capture the process's network traffic for analysis, since the structured JSON prompts and model responses — if recoverable — provide direct evidence of the malware's decision history in a way traditional C2 logs from attacker infrastructure would not.

Short-Term Mitigations

Over the coming quarter, organizations should extend their AI-usage inventory and egress-monitoring programs — many of which were built to govern employee use of AI tools — to also serve a threat-detection function, since the same telemetry that identifies unsanctioned AI tool usage (outbound calls to known model-provider domains) is directly applicable to detecting CLOSEDQUORUM-style malware. DNS and proxy logging for AI-provider domains should be retained and queryable by source process, not just source host, since distinguishing a legitimate developer tool from a malicious implant calling the same API endpoint depends on knowing which process made the call.

Detection engineering teams should build and test hunt queries against the specific technique combinations Talos documented: LSASS access via SeDebugPrivilege and MiniDumpWriteDump, Early Bird APC injection into suspended processes, WMI event subscriptions with 60-second re-trigger intervals, and ETW suppression through EtwEventWrite modification. None of these are new techniques, but their co-occurrence with AI-provider network traffic is the differentiating signal.

Strategic Considerations

Over a longer horizon, this disclosure argues for treating "AI provider API call from an unexpected process" as a first-class detection category alongside traditional C2 beaconing, rather than as a niche signature specific to one malware family. As more malware developers adopt commercial or open-weight models as decision engines — an early, concrete instance of a pattern rather than a hypothetical one — signature-based detection built around specific provider domains or prompt structures is likely to have a short useful life, while behavioral detection built around the pattern of multi-provider AI queries combined with offensive technique execution should generalize better across future variants.

Organizations should also consider what CLOSEDQUORUM implies for their own AI governance posture. The malware's system prompt and JSON-constrained output schema are, in effect, a simple example of prompt engineering used to bypass a model provider's intended use policies for offensive tasks; enterprises building their own agentic tooling on the same commercial model APIs should assume that constrained, schema-bound prompting is an established technique for eliciting narrowly scoped but still harmful outputs from general-purpose models, and should factor that into vendor risk assessments and internal red-teaming of AI-integrated tools.

CSA Resource Alignment

CLOSEDQUORUM is best understood as an agentic AI threat operating largely outside the boundary of an enterprise's own systems, but the underlying architectural pattern — an AI decision-making layer directing tool invocation and action selection based on environmental context — is a close match for the threat categories CSA's MAESTRO framework was designed to model for agentic systems generally. In this note's assessment, MAESTRO's foundation-model and agent-framework layers correspond to CLOSEDQUORUM's structure: the four commercial models function as the foundation-model layer, and the vote-tallying, tie-breaking, and retry logic in the Go binary function as a lightweight agent framework orchestrating those models toward a bounded set of tool actions [7]. Security teams evaluating their own agentic AI deployments can use the same layered decomposition to reason about where a compromised or manipulated decision layer could similarly be steered toward unintended actions, even in defensive or business tooling that has no relationship to malware.

The AI Controls Matrix (AICM) v1.1 provides the operational complement, with control domains covering AI supply-chain integrity, threat and vulnerability management, and logging and monitoring that are directly relevant to the detection posture this research note recommends [8]. AICM's emphasis on monitoring AI-interaction surfaces — logging and reviewing calls an organization's own systems make to model-provider APIs — is the same operational discipline that, applied outward rather than inward, is what allows defenders to notice an unauthorized process making structurally similar calls to multiple AI providers in the CLOSEDQUORUM pattern. Organizations that have already inventoried their legitimate AI-provider API usage under AICM guidance are better positioned to distinguish that baseline from the anomalous multi-provider traffic this malware generates.

More broadly, CLOSEDQUORUM is a concrete instance of a threat category CSA's AI Safety Initiative has tracked across recent disclosures: malware and attack tooling that invoke large language models at runtime rather than shipping fixed logic, which collectively erode the value of static indicators of compromise and push detection toward behavioral and process-identity signals. Enterprises building threat models around agentic AI, whether defensive or as a supply-chain risk from vendors, should treat CLOSEDQUORUM as validating evidence that the LLM-as-decision-engine pattern has moved from research discussion to field-deployed malware, and should prioritize the process-level, multi-provider correlation detections described above accordingly.

References

[1] Cisco Talos. "The CLOSED QUORUM: Inside the First Reported Autonomous AI C2 Implant." Cisco Talos Intelligence Group, September 2026.

[2] The Hacker News. "Windows Malware Is Built to Let Up to 4 AI Models Vote on What It Does Next." The Hacker News, September 22, 2026.

[3] Help Net Security. "CAIRN: Open-Source Framework Helps Identify AI-Powered Malware, Including CLOSEDQUORUM." Help Net Security, September 22, 2026.

[4] Google Cloud / Google Threat Intelligence Group. "GTIG AI Threat Tracker: Advances in Threat Actor Usage of AI Tools." Google Cloud Blog, November 5, 2025.

[5] Anton Cherepanov and Peter Strýček. "First Known AI-Powered Ransomware Uncovered by ESET Research." WeLiveSecurity (ESET), August 26, 2025.

[6] CyberScoop. "NYU Team Behind AI-Powered Malware Dubbed 'PromptLock'." CyberScoop, 2025.

[7] Cloud Security Alliance. "Agentic AI Threat Modeling Framework: MAESTRO." CSA, February 2025.

[8] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." CSA, 2026.