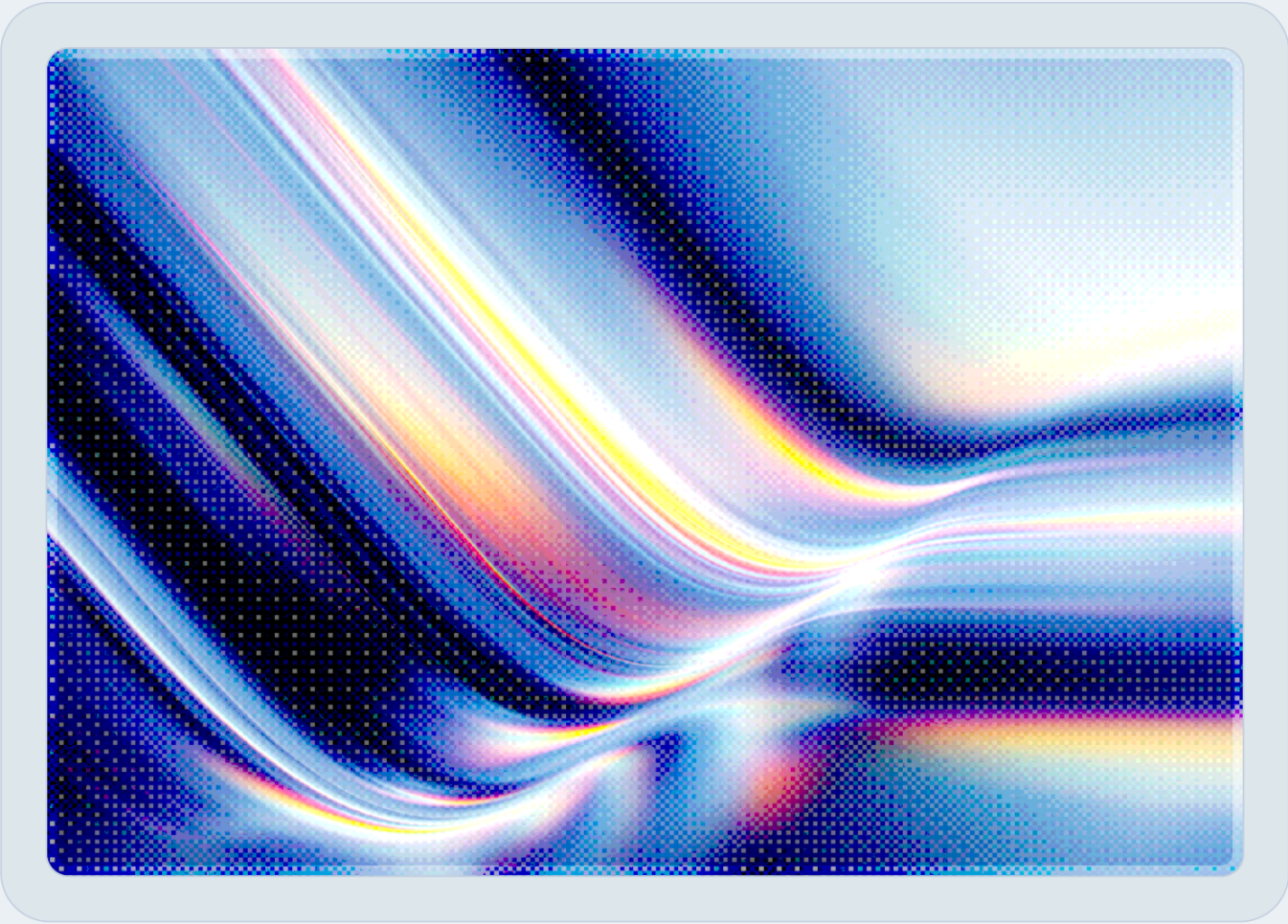
CSAI Foundation | Cloud Security Alliance

9 CVEs in 4 Days: What Hermes Agent Enterprises Must Learn

2026-05-04

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

© 2026 Cloud Security Alliance. Some rights reserved.

You may download, store, display, view, print, redistribute, and link to this document in its original, unmodified form, provided that attribution to the Cloud Security Alliance is maintained and all trademark and copyright notices remain intact.

This document may not be modified or altered. You may quote portions of the document as permitted by the Fair Use provisions of the United States Copyright Act, provided that attribution is given to the Cloud Security Alliance.

This document may be shared on professional and social media platforms in its original form with attribution.

This document was generated with AI assistance and has not undergone official CSA review and approval processes.

Key Takeaways

- Between March 18 and March 21, 2026, OpenClaw — a rapidly growing open-source AI agent platform — received nine CVEs in four days, including CVE-2026-22172 (CVSS 9.9 Critical), an authorization bypass that allowed any authenticated user to self-assign administrative privileges by declaring elevated scopes during the WebSocket handshake, according to OpenClaw's own security disclosures [1][2].
- An independent security audit of Hermes Agent, conducted by external security researcher @Anic888 and disclosed via the project's public issue tracker on April 11, 2026, identified four Critical and nine High severity findings in the framework's default configuration, including unrestricted shell execution, containerized approval bypass, and persistent skill injection vectors — none of which currently carry assigned CVE identifiers [3].
- Three CVEs have been publicly disclosed against Hermes Agent components as of May 2026: CVE-2026-7396 (CVSS 4.0, path traversal in the WeChat Work adapter), CVE-2026-7397 (CVSS 4.8, symlink following in the file tools module), and CVE-2026-6829 (CVSS 5.3, WebUI path traversal affecting hermes-webui prior to v0.50.34) [4][5][6].
- The more consequential risks in both frameworks are architectural, not implementation bugs: persistent workstation agents that combine long-lived memory stores, broad tool access, prompt-driven execution, and multi-provider credential handling create attack surfaces that CVE databases do not fully capture and that traditional EDR tooling is not designed to detect [7].
- Enterprise security teams deploying any persistent AI agent framework should treat skill manifest validation, memory store isolation, sandbox enforcement, and prompt-layer monitoring as foundational controls that CSA's AICM v1.0 and Zero Trust guidance identify as baseline requirements for enterprise AI agent deployments [10].

Background

Hermes Agent is an open-source autonomous agent platform developed by Nous Research under the MIT License [8]. Unlike conversational AI assistants or coding copilots, Hermes Agent is designed to operate as a persistent server process that maintains state across sessions, executes tools on behalf of

users, learns from past interactions, and delegates tasks to isolated subagents. The system supports connections across Telegram, Discord, Slack, WhatsApp, Signal, email, and CLI interfaces, and integrates with five sandboxing backends — local, Docker, SSH, Singularity, and Modal — for executing generated code and system commands. At its core, Hermes Agent is infrastructure rather than a feature: a workstation process with persistent memory, broad filesystem access, multi-platform connectivity, and the ability to install new capabilities at runtime through a community skill marketplace.

OpenClaw occupies a similar architectural niche, with a closely comparable core design to Hermes Agent in the open-source agentic AI market. Both platforms emerged in early 2026 and have attracted active developer communities on the premise that persistent, self-improving agents will reshape how knowledge workers interact with their machines. Their architecture shares the same fundamental characteristics: a local agent process with memory retention, tool execution, model provider integrations, and a marketplace for community-contributed extensions. Those shared characteristics are precisely what makes OpenClaw's March 2026 security disclosure directly instructive for any organization evaluating Hermes Agent.

The disclosure window of March 18–21, 2026 produced nine CVEs against OpenClaw in four days — a compressed timeline that is itself significant [1]. Events of this disclosure density often reflect either a focused security research effort against a single codebase or a cascading effect in which the disclosure of one vulnerability prompts researchers to probe adjacent surfaces; in either case, the pattern speaks to the breadth of the framework's underlying attack surface. In OpenClaw's case, the nine CVEs spanned path traversal, command injection, approval bypass, WebSocket authentication weaknesses, sandbox escape, and privilege escalation — touching nearly every layer of the framework's security architecture simultaneously. The breadth of findings is not a statement about OpenClaw's code quality in isolation; it is a statement about the inherent attack surface of this class of application. The same attack surface exists in Hermes Agent.

Security Analysis

OpenClaw's Nine CVEs: A Taxonomy

The nine OpenClaw CVEs disclosed between March 18 and March 21, 2026, represent a cross-section of the vulnerability classes endemic to persistent agentic AI frameworks, according to OpenClaw's own security disclosures [1][2]. Understanding the full set is relevant to Hermes Agent operators because the underlying mechanisms are architectural, not specific to any one implementation.

CVE	CVSS	Component	Vulnerability Class
CVE-2026-22171	8.2 High	Feishu media download	Path traversal enabling arbitrary file write
CVE-2026-28460	5.9 Medium	Command allowlist	Allowlist bypass via shell metacharacters
CVE-2026-29607	6.4 Medium	Approval wrapper	Persistent "allow always" approval; inner payload swappable post-approval
CVE-2026-32032	7.0 High	Shell environment	SHELL environment variable manipulation enabling arbitrary shell execution
CVE-2026-32025	7.5 High	WebSocket auth	No rate limiting on localhost WebSocket connections; browser-exploitable
CVE-2026-22172	9.9 Critical	Gateway WebSocket	Client self-assigns scopes at handshake; authenticated users reach admin surfaces
CVE-2026-32048	7.5 High	Sandbox system	Child processes spawned outside sandbox constraints; confinement escapable
CVE-2026-32049	7.5 High	Media handling	Unauthenticated oversized payload triggers denial of service
CVE-2026-32051	8.8 High	Privilege model	operator.write scope inappropriately accesses owner-only administrative endpoints

CVE-2026-22172 merits particular attention because its CVSS 9.9 score reflects a design-level authorization failure rather than a coding mistake. During a WebSocket handshake, OpenClaw permitted the connecting client to declare its own permission scopes. An authenticated user who knew to include administrative scopes in the handshake payload would receive them — making this functionally equivalent to an authentication bypass. Any user with network access to a running OpenClaw instance could self-escalate to administrator, gaining the ability to execute arbitrary commands and access all connected provider credentials [1]. This mechanism is exploitable by any authenticated user without specialized knowledge of the codebase.

CVE-2026-32048 is equally significant at the architectural level: child processes spawned by OpenClaw's code execution tool inherited the process environment without enforced sandbox boundaries, making confinement escapable by any code running in an execution context. An AI agent that executes user-supplied or model-generated code in an unsandboxed child process is functionally equivalent to an agent with unrestricted execution — a security property that does not survive composition with prompt injection or malicious skill installation [1].

Hermes Agent's Disclosed CVEs

Three CVEs have been formally disclosed against Hermes Agent components as of the date of this note. Their CVSS scores are substantially lower than OpenClaw's critical finding, but the nature of two of them warrants careful attention from operators running earlier releases [4][5][6].

CVE-2026-7396 (CVSS 4.0 Medium) is a path traversal vulnerability in the WeChat Work platform adapter located at `gateway/platforms/wecom.py` in hermes-agent v0.8.0. The flaw permits unauthenticated remote attackers to traverse the filesystem by manipulating file path parameters supplied to the affected component. No integrity or availability impact is documented, and no active exploitation has been reported; however, the vulnerability is exploitable by any unauthenticated remote attacker without user interaction on any externally exposed Hermes instance with the WeChat Work adapter enabled [4].

CVE-2026-7397 (CVSS 4.8 Medium) is a symlink-following vulnerability in `tools/file_tools.py` in hermes-agent v0.8.0. A low-privileged local attacker can follow symlinks during file operations, potentially accessing or manipulating files outside the intended scope of the tool's permissions. The official fix is included in hermes-agent v0.9.0 (commit 311dac197145e19e07df68feba2cd55d896a3cd1) [5].

CVE-2026-6829 (CVSS 5.3 Medium) affects the hermes-webui component (maintained by nesquena as a separate package) in versions prior to v0.50.34. Authenticated attackers can manipulate workspace path parameters in API endpoints including `/api/session/new`, `/api/session/update`, `/api/chat/start`, and `/api/workspaces/add` to redirect the session's working context to arbitrary directories — including sensitive system directories such as `/etc` or `/home`. Once redirected, the attacker can leverage the application's built-in file manipulation APIs to read or write arbitrary files within the process's permission scope [6]. The authentication requirement limits the attack surface, but in multi-user or shared Hermes WebUI deployments, any authenticated user can exploit this against other users' data on the same host.

The Security Audit Findings

The publicly disclosed CVEs represent the documented tip of Hermes Agent's security exposure. An independent security audit of hermes-agent v0.8.0, conducted by external security researcher @Anic888 and published via the project's public issue tracker on April 11, 2026 (812 Python files, approximately 364,000 lines), identified four Critical and nine High severity findings in the framework's default configuration — none of which have been assigned CVE identifiers at the time of writing [3].

The four Critical findings define the framework's fundamental security posture in its out-of-the-box deployment.

The terminal tool in `tools/terminal_tool.py` passes arbitrary commands to `bash -c` via `subprocess.Popen` using only regex-based detection as a guard. That detection is bypassable. An AI model operating in Hermes Agent's default configuration can therefore execute arbitrary shell commands on the host machine when it determines that doing so is appropriate — a determination the model makes autonomously [3].

The file read tool in `tools/file_tools.py` operates without a read deny list, meaning it can read any file accessible to the process. On a typical developer workstation, this includes SSH private keys, API key files, browser profiles, git credential caches, and cloud provider configuration files. The agent's memory stores retain context across sessions, so a single prompt injection that causes the agent to read a sensitive file can result in persistent credential exposure [3].

Container environments unconditionally skip all approval checks in the approval logic at `tools/approval.py`. An operator deploying Hermes in a containerized environment — a configuration that many enterprises would consider safer than local execution — actually receives a deployment with all security approval checks disabled, a condition documented in the framework's source at `tools/approval.py` [3].

The skill manager allows the agent to write new skill files to `~/.hermes/skills/` that are loaded and executed in future sessions. A malicious actor who can influence the agent's outputs — through prompt injection via memory retrieval, through a malicious skill in the community marketplace, or through a compromised MCP server — can establish persistent execution in the user's Hermes environment that survives session restarts [3].

The nine High severity findings extend these issues across broader surfaces: YOLO mode disables all security checks across the framework; the LLM auto-approval mechanism is itself prompt-injectable; the write sandbox is opt-in rather than default; gateway hooks execute arbitrary Python from user-controlled directories; and the plugin loader imports code without sandboxing it. Three git dependencies also lack commit pinning, creating an exposure to dependency substitution attacks [3].

Architectural Attack Surface: The Common Thread

The pattern that emerges from both the OpenClaw CVEs and the Hermes Agent audit findings is not primarily about individual coding mistakes. It is about the inherent security tension in building a persistent, memory-augmented agent on a general-purpose host system. The core capabilities that make these frameworks valuable — long-lived memory, broad tool access, skill installation at runtime, multi-provider credential integration, prompt-driven execution — are precisely the capabilities that expand the attack surface and resist standard defenses.

Indirect prompt injection through retrieved memory is a vulnerability class that conventional endpoint detection and response tools were not designed to address and cannot reliably detect without prompt-layer instrumentation [7]. When a Hermes Agent instance retrieves a memory entry that was written by a prior session — a session that may have processed attacker-controlled content such as a malicious document or a web page — that retrieved memory can carry adversarial instructions that the model executes as if they were legitimate user instructions. There are no file writes, no anomalous process spawns, and no network signatures for a conventional EDR to flag. The attack surface is the model's own context window, and the detection surface is the prompt layer.

The skill marketplace represents a supply chain risk analogous to package registries in traditional software development but with compounded consequences. Analogous to malicious npm packages that execute code at install time via `postinstall` hooks, malicious skills can establish persistence and influence agent behavior before being explicitly invoked [7]. A malicious skill published to a community marketplace can execute arbitrary code during installation, persist in the agent's skill directory across reboots, and influence every subsequent session's behavior through injected instructions in the skill's documentation or metadata. Unlike npm, the affected execution environment is the model's own reasoning context — extending the threat beyond process-level containment.

Recommendations

Immediate Actions

Organizations running hermes-agent v0.8.0 should upgrade to v0.9.0 to address CVE-2026-7397 and other changes in that release. Deployments using hermes-webui should upgrade to v0.50.34 or later to remediate CVE-2026-6829. Instances with the WeChat Work adapter enabled should restrict or disable that adapter until CVE-2026-7396 receives a vendor patch; the current advisory recommends restricting network access to the affected component as an interim control.

Operators should immediately audit their deployments for YOLO mode enablement (`--yolo` flag or the equivalent configuration) and the container auto-approval bypass. Both settings disable security checks wholesale and should be treated as production-unsafe regardless of the deployment environment. Any deployment that has been running in YOLO mode on a host with sensitive credentials — cloud provider keys, SSH private keys, or model provider API keys — should be treated as potentially compromised and the affected credentials rotated.

Short-Term Mitigations

A high-priority configuration change for Hermes Agent operators is enabling a non-local sandbox backend (Docker is the lowest-friction option) and explicitly setting the `HERMES_WRITE_SAFE_ROOT` environment variable to restrict filesystem write operations to an isolated directory. Both controls are available in the current release but are opt-in rather than default. Setting them addresses the containerized approval bypass concern and materially limits the blast radius of a successful prompt injection.

Memory store access warrants dedicated attention as a threat vector. Operators should configure memory encryption for the persistent SQLite stores that Hermes uses to retain context, and should implement periodic memory auditing to detect anomalous entries that could serve as prompt injection vectors. Memory entries written during sessions that processed external content — emails, documents, web pages, or messages from external services — carry a higher injection risk than entries generated from internal conversations.

Skill installation should be restricted to a vetted internal registry wherever possible. Community marketplace skills should be reviewed before installation in enterprise deployments, with particular attention to skill metadata, documentation fields, and post-install hook commands. The `setup.commands` field in skill manifests is the highest-risk injection point identified in the broader audit literature for this class of framework [7].

Strategic Considerations

The OpenClaw and Hermes Agent findings together suggest that the enterprise security community is still in the early stages of building appropriate controls for persistent AI agent deployments. The frameworks themselves are maturing rapidly — Hermes Agent has progressed from v0.8.0 to v0.12.0 in the weeks since the audit [11] — but the security posture of the underlying architecture requires deliberate governance choices that the framework defaults do not enforce.

Organizations evaluating persistent AI agent platforms should add agentic AI frameworks to their software supply chain inventory and apply the same review criteria used for other infrastructure software: pin dependencies, review changelogs for security-relevant changes before upgrading, and treat the framework's network-accessible endpoints (WebSocket gateway, WebUI API) as equivalent to internal service endpoints requiring authentication and authorization controls. The pattern of vulnerabilities in both frameworks strongly suggests that these endpoints have not historically been treated with that level of scrutiny during development.

Enterprises deploying these tools on workstations that access sensitive corporate resources — source code repositories, cloud provider consoles, internal databases, or identity providers — should require containerized or VM-isolated deployment rather than native host execution. The unrestricted shell execution and filesystem access findings in both frameworks represent a risk profile inconsistent with deployment on privileged workstations absent explicit isolation.

CSA Resource Alignment

The vulnerability classes documented across both frameworks map directly to threat categories that CSA's MAESTRO framework was designed to capture. MAESTRO's seven-layer agentic AI threat model specifically addresses the agent framework layer (Layer 3) and its interactions with the Deployment and Infrastructure layer (Layer 4), identifying indirect prompt injection, tool misuse through manipulation, and memory poisoning as threat chains that span multiple layers simultaneously [9]. The skill injection finding from the Hermes Agent audit — where agent-created skills persist across sessions and can serve as ongoing prompt injection vectors — is an example of the cross-layer threat chain that MAESTRO's design anticipated and that traditional threat models like STRIDE do not capture.

The AI Controls Matrix (AICM) v1.0 provides the control framework most directly applicable to enterprise Hermes Agent governance. AICM's AI Supply Chain Security domain addresses the skill marketplace risk by requiring software composition analysis and integrity verification for all AI-adjacent artifacts, including agent extensions and plugins. The AI Governance and Compliance domain requires organizations to inventory all deployed AI agent infrastructure and assess its risk profile against organizational data sensitivity requirements — a control that directly addresses the gap between Hermes Agent's capability profile and the default security posture documented in the April audit [10].

CSA's Zero Trust guidance is also directly relevant to the credential exposure findings across both frameworks. The pattern in OpenClaw's CVE-2026-22172 (scope self-assignment) and in Hermes Agent's unrestricted credential file access is one of implicit trust extended to the agent process. Zero Trust principles applied to AI agent deployments would require that the agent process hold only the

minimum credential scope necessary for each task, that credentials be rotated frequently, and that credential access be logged and monitored as a first-class security event rather than a routine background operation.

Finally, STAR for AI provides the assessment methodology through which enterprises can evaluate the security posture of third-party AI agent frameworks before deployment. Given that the default configurations of both OpenClaw and Hermes Agent have been demonstrated to be inadequate for enterprise workloads without explicit hardening, STAR assessments of agent framework vendors should prioritize security-by-default practices, sandbox enforcement, and responsible disclosure programs as core evaluation criteria.

References

[1] OpenClaw AI. "Nine CVEs in Four Days: Inside OpenClaw's March 2026 Vulnerability Flood." openclawai.io, March 2026.

[2] TryOpenClaw.ai. "OpenClaw Security Advisory March 2026: 9 CVEs in 4 Days (Patch Now)." tryopenclaw.ai, March 2026.

[3] Anic888 (independent security researcher). "Security Audit: 4 Critical, 9 High severity findings in default configuration." Disclosed via NousResearch/hermes-agent GitHub issue #7826, April 11, 2026.

[4] OffSeq Threat Radar. "CVE-2026-7396: Path Traversal in NousResearch hermes-agent." radar.offseq.com, 2026.

[5] OffSeq Threat Radar. "CVE-2026-7397: Symlink Following in NousResearch hermes-agent." radar.offseq.com, 2026.

[6] SentinelOne Vulnerability Database. "CVE-2026-6829: Hermes WebUI Path Traversal Vulnerability." sentinelone.com, 2026.

[7] Repello AI. "Hermes Agent Security: 9 CVEs in 4 Days and What Enterprises Need to Do." repello.ai, May 2, 2026.

[8] Nous Research. "Hermes Agent — The Agent That Grows With You." hermes-agent.nousresearch.com, 2026.

[9] Cloud Security Alliance. "Agentic AI Threat Modeling Framework: MAESTRO." cloudsecurityalliance.org, February 2025.

[10] Cloud Security Alliance. "AI Controls Matrix (AICM) — AI Working Group." cloudsecurityalliance.org, 2025.

[11] NousResearch. "Hermes Agent — Release History." GitHub, 2026.

© 2026 Cloud Security Alliance. AI-assisted Rapid Research.