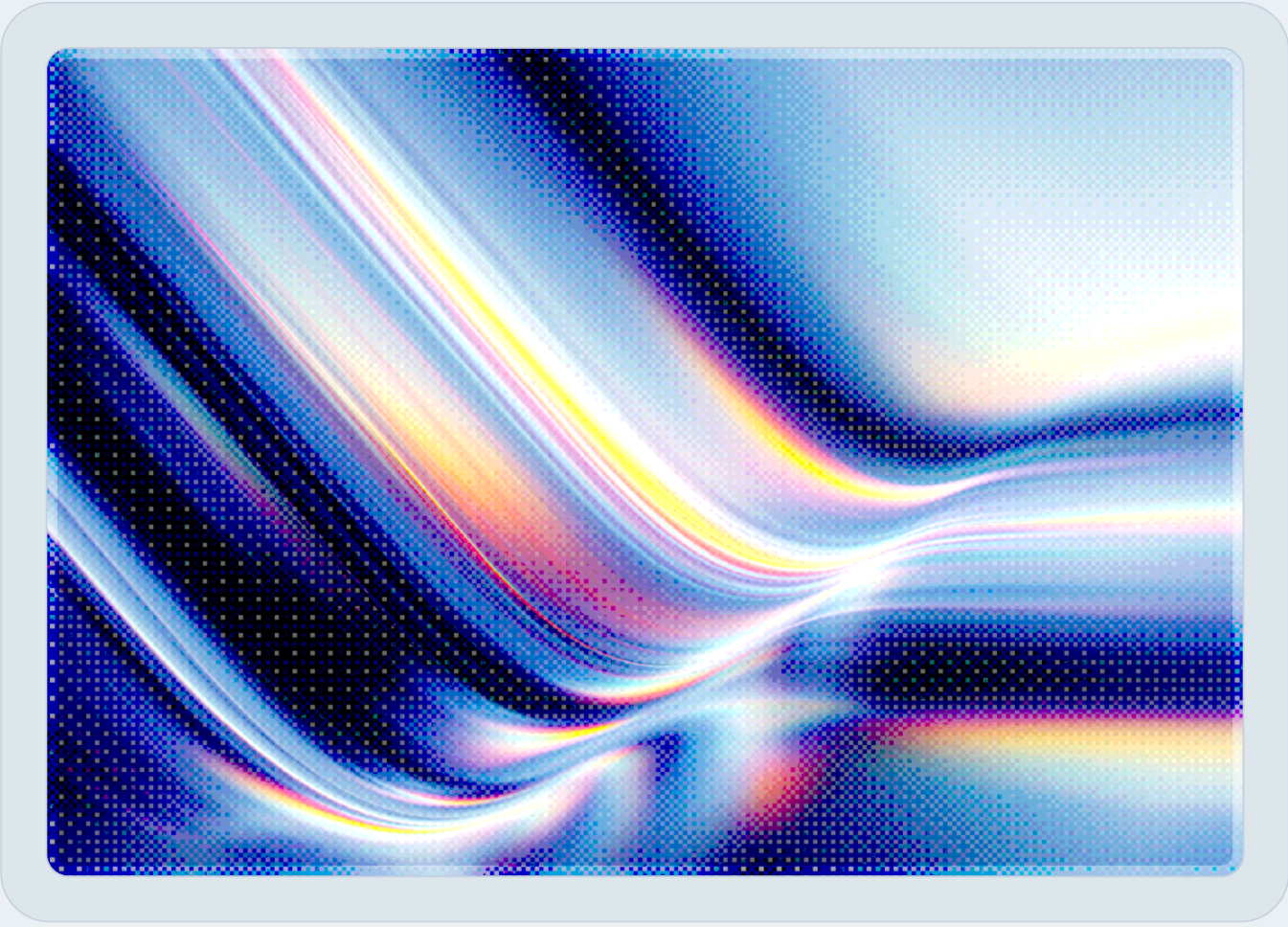
CSAI Foundation | Cloud Security Alliance

AI Memory, Compromised: The MemTensor Supply Chain Attack

2026-09-25

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

On September 23, 2026, an attacker compromised the GitHub Actions release pipelines of MemTensor, publisher of the widely used MemoryOS and OpenClaw memory-integration tooling, and used stolen publishing tokens to push credential-stealing builds to npm and PyPI [1][2]. The malicious releases embedded a cross-platform Go implant known as sckit inside packages that AI agents load to persist and recall long-term memory, meaning the compromise reached developer machines and CI runners through a dependency that security teams have not historically prioritized for scrutiny [3]. Because the implant activated during normal package import and agent operation rather than through a conspicuous install script, it evaded install-time scrutiny, the stage where many dependency-scanning tools concentrate detection effort [4]. The incident also exposed a structural risk specific to AI agent architecture: because MemTensor's OpenClaw plugin forwarded prompt text through the same code path the implant hooked, any secrets a user typed into an agent conversation while the plugin was active must now be treated as compromised [3][5]. This research note examines the attack chain, the credentials and data at risk, and the practical steps organizations running AI memory infrastructure should take in response.

Background

MemTensor develops open-source tooling for giving large language model agents persistent, long-term memory, most notably the MemoryOS library distributed on PyPI and a companion plugin, published under the npm scope `@memtensor`, that connects MemoryOS to OpenClaw, a widely adopted agentic AI runtime [2][6]. Memory subsystems of this kind have become increasingly common in agent stacks, reflecting a broader shift toward agents that retain context, prior decisions, and user preferences across sessions rather than starting from a blank slate each time they are invoked. That utility comes with a corresponding increase in attack surface: a memory plugin sits directly on the path between a running agent, the credentials available to its host process, and the raw content of the prompts it exchanges with users.

On September 23, 2026, between roughly 00:48 and 05:25 UTC, an unknown actor published malicious versions of MemoryOS on PyPI (version 2.0.34) and of the npm package `@memtensor/memos-cloud-openclaw-plugin` (versions 0.1.21, 0.1.23, and 0.1.25) [4][7]. Independent researchers first

flagged the discrepancy hours later, and the blockchain security firm SlowMist issued a public warning about the compromise on September 24 that was quickly corroborated by multiple supply chain security vendors, including StepSecurity, SafeDep, and Semgrep [1][3][4]. PyPI subsequently quarantined the malicious MemoryOS release, and clean versions were restored on both registries, with npm 0.1.20 and PyPI 2.0.33 identified as the last known-good builds before the compromise [4].

The attacker did not exploit a vulnerability in MemoryOS or OpenClaw itself. Instead, they obtained MemTensor's own npm and PyPI publishing tokens by manipulating the organization's GitHub Actions release workflows, a technique that has recurred across several AI-ecosystem supply chain incidents in 2026 and that CSA has tracked in prior research on npm-focused campaigns [8]. According to technical analysis published by SafeDep, the attacker's commits caused the release job to hand over its publishing token before the legitimate publish step ran, in one case by writing to `$GITHUB_ENV` in a way that let a `BASH_ENV`-triggered script execute ahead of the real `npm publish` command, and in the PyPI case by using a similar `BASH_ENV`-injected bridge script to intercept the publish step and capture the token before it completed [4][9]. Because the resulting packages were published from MemTensor's legitimate, previously trusted accounts, they carried little to no signal that would have distinguished them from an ordinary release to a downstream user or an automated scanner relying on publisher reputation alone.

This absence of a distinguishing signal is what makes CI/CD token theft harder to detect than more conventional approaches, such as account takeover through credential stuffing or typosquatting a similarly named package. A typosquat or a lookalike package name can be caught by a vigilant developer reading a dependency tree, and an account takeover through a phished password can in principle be caught by monitoring for anomalous login activity. A release built and signed through the legitimate CI/CD pipeline of the actual maintainer, published under the actual package name, carries none of those tells; it is functionally indistinguishable from a routine version bump until someone inspects the code the release actually ships. This may help explain why the MemTensor incident, like the CI/CD-focused campaigns CSA has previously documented, was caught by external researchers monitoring package registries for behavioral anomalies rather than by any control internal to the affected organization [4][8].

Security Analysis

The implant delivered through both compromised packages is a statically linked, cross-platform Go binary that researchers have named sckit, built to run on Windows, Linux, and macOS [1][3]. Its design avoids the detection patterns most commonly used against supply chain malware. Rather than executing through a `preinstall` or `postinstall` hook, which security tooling and manual code review increasingly flag as a high-risk pattern, the npm package imported a function that launched the payload

during normal plugin initialization, specifically when the OpenClaw agent gateway started or when the plugin handled a memory-recall event [3][4]. The PyPI package achieved a similar effect by hooking into the library's logging configuration, so that simply importing MemoryOS in a normal Python session triggered the credential-stealing routine [3]. Both mechanisms meant the malicious behavior was reachable through ordinary use of the software rather than through an unusual or auditable installation step.

Once running, sckit systematically inventoried the host's home directory and environment variables for credentials associated with developer and cloud tooling, including npm and PyPI publishing tokens, GitHub and GitLab access tokens, AWS credentials, Hugging Face tokens, HashiCorp Vault tokens, and secrets for services such as Slack, Stripe, and SendGrid, exfiltrating what it found to attacker-controlled infrastructure at domains under `skyleen[.]fr` [1][4]. Static analysis by StepSecurity further identified functions within the binary named in ways consistent with automated re-propagation, including logic to prepare and publish workflow changes to other repositories, along with a conditional helper that activated specifically inside GitHub Actions publishing environments and was positioned to capture tokens such as `PYPI_API_TOKEN` before the workflow could unset them [9]. The presence of this reusable workflow-hijacking logic in a credential-stealing implant differs from the single-package, non-propagating compromises CSA has previously documented in this space [8][9], though StepSecurity found no evidence the logic was actually used to propagate, characterizing this as limited self-propagation potential rather than confirmed worm behavior.

The OpenClaw-specific behavior of the npm package introduces a risk that is distinct from conventional credential theft. Because the plugin's memory-recall path forwarded the user's prompt text through the same code that triggered the implant, any information a developer or user entered into an agent conversation while the compromised plugin was active — including secrets a user might paste into a prompt without expecting it to be logged or transmitted anywhere — should be treated as exposed to the attacker [3][4]. This distinguishes the incident from a conventional developer-tooling compromise, which is generally understood to bound blast radius to credentials present in the build or runtime environment: here, the compromised component sat inside the data path of the AI agent itself, giving the attacker visibility into a channel many teams do not currently threat-model as a credential source.

Taken together, the MemTensor incident is consistent with a pattern CSA has tracked across several 2026 npm and PyPI campaigns, in which attackers target the CI/CD release pipeline of a trusted maintainer rather than attempting to compromise end users directly, thereby inheriting the full trust and reach of the legitimate publisher [8]. What differentiates this case is the target: rather than a general-purpose developer library or an AI framework's core package, the compromised software was purpose-built memory infrastructure for AI agents, a category of dependency that concentrates unusually dense credential access and, in this instance, direct exposure to prompt content.

The specific credential set sckit targeted is also informative. Rather than harvesting a single class of secret, the implant's inventory logic swept broadly across developer tooling, cloud infrastructure, and third-party service credentials in a single pass, consistent with a strategy of broad credential collection rather than targeting a single credential type in advance [1][4]. This is consistent with the broader observation, echoed across CSA's supply chain research through 2026, that AI development environments now concentrate a wide range of credentials in a single place: a workstation or build runner routinely holds package registry tokens, source control access, cloud provider keys, and now, increasingly, the API keys and session data associated with the AI models and agent frameworks themselves [8]. An attacker who successfully lands an implant in that environment does not need to know in advance which credential will prove most valuable; broad collection lets them sort that out after exfiltration.

Recommendations

Immediate Actions

Organizations that installed `@memtensor/memos-cloud-openclaw-plugin` versions 0.1.21, 0.1.23, or 0.1.25, or MemoryOS version 2.0.34 from PyPI, between September 23 and the time malicious versions were pulled from the registries should treat every credential reachable from the affected host as compromised. This includes rotating npm and PyPI publishing tokens, GitHub and GitLab access tokens, cloud provider keys (AWS, and any other credentials present in the environment), Hugging Face tokens, HashiCorp Vault tokens, and application secrets such as Slack, Stripe, or SendGrid keys that may have resided on the same system [1][4]. Teams should also search process telemetry and network logs for connections to `skyleen[.]fr` and its subdomains, terminate any surviving sckit processes, and downgrade to the last known-good releases, npm version 0.1.20 and PyPI version 2.0.33, or a later clean release if one is available [4][9]. Because the npm variant exposed recalled prompt text to the implant, organizations should also review what sensitive information may have passed through agent prompts while the compromised plugin was running and notify affected users or customers where that exposure is material.

Short-Term Mitigations

Beyond immediate remediation, organizations should audit their CI/CD release workflows for the class of vulnerability that enabled this compromise: environment variables or build hooks (such as `BASH_ENV` or a custom build backend) that can execute attacker-supplied code before a publishing step runs and inherit that step's token in the process. Reviewing GitHub Actions workflows for publishing jobs,

restricting token scope and lifetime, and requiring multi-party review before changes to release pipelines take effect would likely have made this specific technique — hijacking the token hand-off before the publish step — significantly harder to execute. Security teams should also extend software composition analysis and dependency monitoring to explicitly cover AI memory and agent-integration packages, which have not historically received the same scrutiny as core application dependencies but now sit on a path with access to both infrastructure credentials and live agent prompt content.

Strategic Considerations

The MemTensor incident illustrates that AI agent memory subsystems represent a supply chain target distinct from the AI frameworks and coding-agent tooling that CSA's prior research has focused on [8], and one that combines credential exposure with direct access to prompt content. Any component that persists agent state, recalls prior context, or otherwise sits in the data path between a user's prompt and the AI system's behavior should be inventoried and governed with the same rigor organizations apply to authentication and secrets-management infrastructure, because a compromise there can expose both credentials and raw conversational content simultaneously. The AI Controls Matrix's Supply Chain Management, Transparency, & Accountability domain, discussed in CSA Resource Alignment below, offers a concrete starting point for that governance work. Organizations building or procuring agentic AI systems should require vendors to disclose their CI/CD publishing controls, including whether release tokens are scoped, short-lived, and protected against the injection patterns seen in this and comparable 2026 incidents, as part of ordinary vendor risk assessment.

CSA Resource Alignment

This incident reinforces findings in CSA's npm Supply Chain Under Siege: TeamPCP, Miasma, and npm v12, which documented the same underlying technique — attackers hijacking a legitimate publisher's GitHub Actions release pipeline to obtain a signed, trusted publishing token rather than attacking end users directly. That paper's analysis of OIDC token theft, lifecycle-hook exploitation, and the limits of provenance attestation applies directly to the MemTensor compromise and should inform any CI/CD hardening undertaken in response. CSA's The Developer Toolchain as Enterprise Attack Surface similarly documents CI/CD pipelines as holding a concentration of build, signing, and publishing secrets that makes them high-value targets, a framing directly applicable to how MemTensor's release pipeline was compromised [10]. Because the compromised plugin specifically targeted the OpenClaw agent runtime, organizations should also consult CSA's OpenClaw Security Hardening Guide for configuration and isolation guidance applicable to any OpenClaw deployment that may have run the affected memory plugin. The incident is also a close analogue to CSA's prior research note on the Sapphire Sleet

compromise of the Mastra AI npm supply chain, in which attackers similarly targeted an AI agent framework's package ecosystem to reach developer credentials at scale; the recurrence of this pattern against a second category of AI infrastructure within months suggests it reflects a durable attacker strategy rather than an isolated event. Finally, organizations assessing their overall exposure should map this incident to the Supply Chain Management, Transparency, & Accountability domain of CSA's AI Controls Matrix (AICM) v1.1, which provides controls for third-party AI component vetting, build pipeline integrity, and credential governance directly relevant to preventing and containing incidents of this kind.

References

[1] StepSecurity. "Sckit Supply Chain Worm Hits MemTensor npm & PyPi scopes." StepSecurity Blog, September 2026.

[2] The Hacker News. "Compromised MemTensor Packages Deliver sckit Credential Stealer via npm and PyPI." The Hacker News, September 2026.

[3] Semgrep. "AI Supply Chain Attack Hits an OpenClaw Memory Plugin." Semgrep Blog, September 2026.

[4] SafeDep. "MemTensor npm and PyPI Packages Hit by a Go Worm." SafeDep, September 2026.

[5] KuCoin. "MemTensor AI memory component compromised; malicious code executed via PyPI/npm packages." KuCoin News Flash, September 2026.

[6] KuCoin. "MemTensor AI Memory Tools supply chain attacked; developer credentials at risk." KuCoin News Flash, September 2026.

[7] KuCoin. "SlowMist Warns of Malware in MemTensor AI Memory Component via PyPI/npm Packages." KuCoin News Flash, September 2026.

[8] Cloud Security Alliance. "npm Supply Chain Under Siege: TeamPCP, Miasma, and npm v12." CSA AI Safety Initiative, June 2026.

[9] StepSecurity. "Sckit Supply Chain Worm Hits MemTensor npm & PyPi scopes." StepSecurity Blog, September 2026.

[10] Cloud Security Alliance. "The Developer Toolchain as Enterprise Attack Surface." CSA AI Safety Initiative, May 2026.

