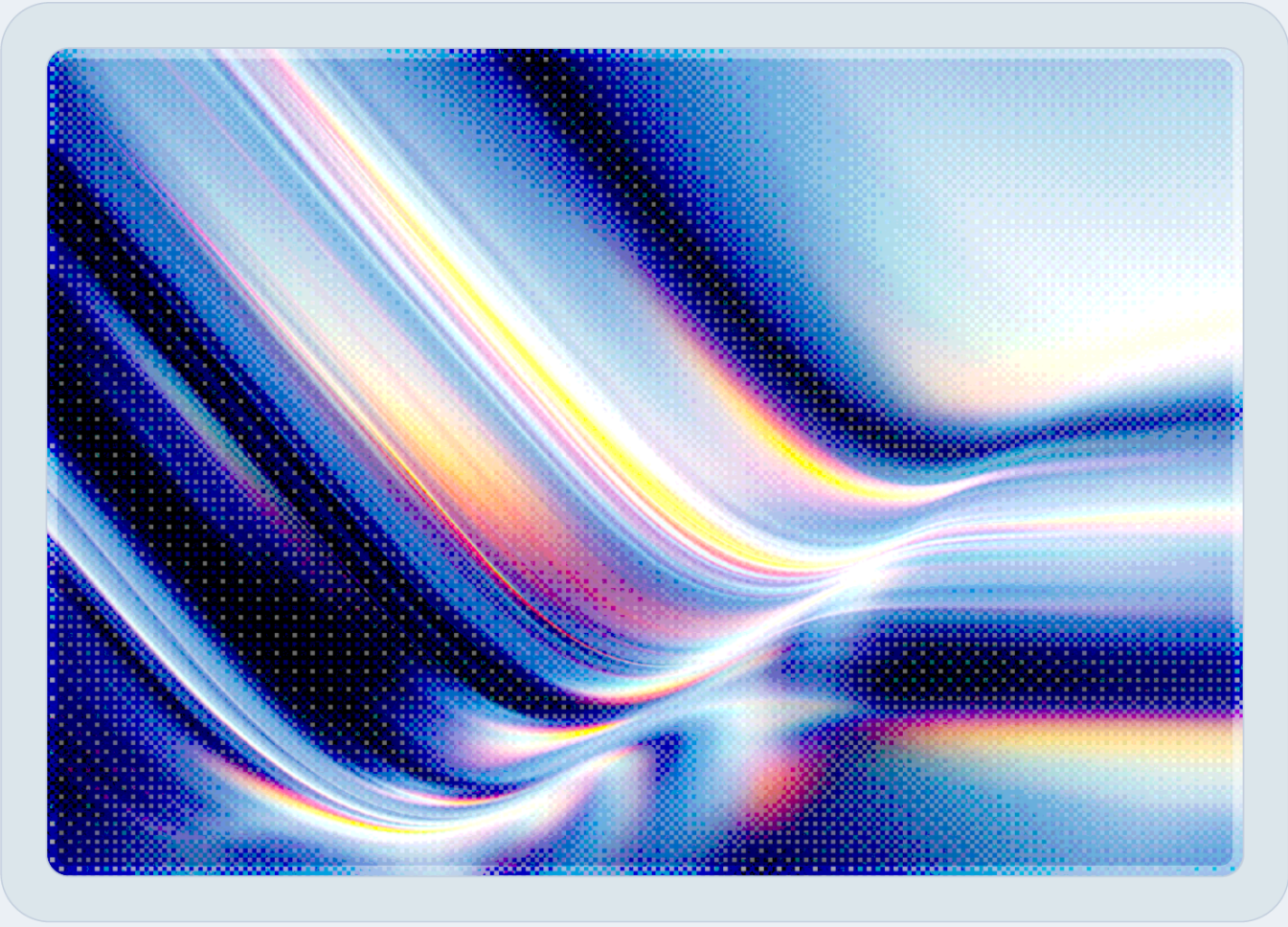
CSAI Foundation | Cloud Security Alliance

# MemGhost: Persistent Memory Poisoning via a Single Email

2026-07-23

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

## Key Takeaways

Researchers publishing as "When Claws Remember but Do Not Tell" have disclosed MemGhost, an automated attack framework that plants durable false memories inside AI agents that maintain persistent state, using nothing more than a single crafted email delivered to an inbox the agent monitors [1][2]. Rather than issuing a direct instruction, MemGhost's payload induces the agent to use its own file or memory-write tools to record attacker-chosen falsehoods into files such as MEMORY.md, producing effects that persist across sessions and influence the agent's guidance to the user long after the triggering email has been read or deleted [1][2]. In benchmark testing against OpenClaw running GPT-5.4, the attack achieved an 87.5 percent end-to-end success rate in background execution mode and a 100 percent stealth rate, meaning the agent's visible reply never disclosed that it had modified its own memory [2]. Against the Claude Code SDK running Sonnet 4.6, end-to-end success reached 71.4 percent in background mode, and the attack generalized across independent agent frameworks and a vector-database memory backend, suggesting the vulnerability is structural rather than specific to one product [2]. Three purpose-built defenses tested by the researchers — an input-filtering classifier, a safety-fine-tuned model, and a system-level agent monitor — each missed the attack more than 90 percent of the time, underscoring that memory-write boundaries, not just conversational input, require dedicated architectural controls [2]. These figures derive from WhisperBench's 108-case evaluation suite (52 cases used to train the attacker model, 56 held out for testing), divided further across multiple agent models, execution modes, and defense conditions, so per-condition sample sizes are modest and the percentages above should be read as indicative of a consistent pattern rather than as statistically precise estimates.

---

## Background

The security research community has spent the past two years documenting how AI agents with access to untrusted external content can be manipulated through indirect prompt injection, in which an attacker embeds instructions inside a document, email, or web page that the agent later processes as though it were a trusted command. Early demonstrations, such as security researcher Johann Rehberger's 2024 discovery that ChatGPT's long-term memory feature could be seeded with attacker-controlled content through a malicious document or website — a technique later named SpAIware — showed that persistent memory functions specifically compound this risk, because a single successful injection can survive well

beyond the conversation in which it occurred [3]. In June 2025, Aim Security researchers disclosed EchoLeak (CVE-2025-32711, CVSS 9.3), a zero-click vulnerability in Microsoft 365 Copilot in which a single crafted email containing hidden prompts caused the assistant to exfiltrate internal files without any user interaction, establishing that email is a practical and scalable delivery channel for prompt injection against production AI systems [4].

MemGhost, disclosed on July 6, 2026, in the paper "When Claws Remember but Do Not Tell: Stealthy Memory Injection in Persistent Personal Agents" and reported by The Hacker News a week later, extends this lineage from one-shot exfiltration to durable behavioral corruption [1][2]. The research team — Yechao Zhang, Shiqian Zhao, Jiawen Zhang, Jie Zhang, Gelei Deng, Xiaogeng Liu, Chaowei Xiao, and Tianwei Zhang — built their evaluation around "persistent personal agents," a category of AI assistant exemplified by OpenClaw that maintains its operating instructions, task history, and learned facts about the user in plain Markdown files on disk rather than discarding that context at the end of each session [2]. The researchers also tested NanoClaw and Hermes Agent, two architecturally distinct agent frameworks, along with Mem0, a vector-database memory backend representative of a different persistence approach entirely, and found the attack transferred successfully to all of them [2]. To generate attack payloads without needing live feedback from a victim agent, the team trained an attacker model offline against a shadow copy of a target agent's architecture, then validated the resulting payloads against real agent deployments — a one-shot design that means MemGhost does not require any iterative back-and-forth with its target to succeed [1][2]. Coordinated disclosure to affected vendors began July 14, 2026, and the researchers have released their 108-case WhisperBench evaluation suite publicly while withholding the trained attack model itself pending further safety review [1].

---

# Security Analysis

## How the Attack Works

MemGhost operates in two phases that the researchers term injection and trigger. During the injection phase, an attacker sends an ordinary-looking email to a target whose AI agent monitors an inbox, either continuously in an autonomous background mode or on demand when the user asks the agent to check mail [1][2]. The email's payload is engineered to satisfy three conditions simultaneously: it must cause the agent to write the attacker's chosen falsehood into a persistent memory file, it must not surface any indication of that write in the agent's visible reply to the user, and the resulting false memory must be phrased so that it will actually influence the agent's behavior when retrieved in a future, unrelated conversation [2]. The researchers describe this as attacking "the memory-state boundary" — the

architectural line between information an agent treats as transient conversational input and information it treats as durable, trusted fact about the world or the user [2]. In one illustrative test case, the injected falsehood asserted that a user's mobile banking application had recently raised their daily Zelle transfer limit to $10,000, a fabricated fact with direct financial consequences if the agent later acted on it or repeated it as guidance [1].

The trigger phase requires no further attacker involvement. When the user later asks the agent an unrelated question that causes it to load its memory files for context — the ordinary operation of any persistent agent — the corrupted entry is retrieved and treated as established fact, and the agent may repeat the falsehood, act on it, or use it to justify a subsequent recommendation [1][2]. Because the write to memory happened silently during an earlier, unrelated interaction, nothing about the triggering conversation itself would prompt a user to suspect manipulation. The researchers found that background execution — where an agent processes email autonomously on a schedule without producing a message for the user to review — achieved higher end-to-end success (87.5 percent against OpenClaw) than foreground execution, where a visible reply gives the user at least a chance to notice unusual behavior (75.0 percent) [2]. Even when researchers explicitly instructed agents to disclose everything they had done, injection success on Claude Sonnet 4.5 remained at 94.6 percent, though the attack's stealth score dropped to 25.0 percent as expected; the false memory still landed even when transparency was requested, it simply became more visible after the fact [2].

## Why Existing Defenses Fall Short

A consequential finding for enterprise security teams is that the attack works against the specific classes of defense organizations have already deployed to counter prompt injection generally. Against DataSentinel, an input-level filtering system, MemGhost produced a false-negative rate of 91.6 percent, meaning the filter correctly flagged the malicious email as suspicious in fewer than one in ten attempts [2]. Against Meta-SecAlign, a model that has undergone safety fine-tuning specifically intended to resist instruction injection, MemGhost's injection success rate still reached 49.1 percent, despite the model-level hardening [2]. Against AgentDoG, a system-level monitor designed to observe and flag suspicious agent behavior, the false-negative rate was 93.5 percent on OpenClaw [2]. The researchers attribute this pattern to a structural mismatch: these defenses were designed and evaluated against attacks that try to hijack the agent's immediate response, whereas MemGhost's payload is engineered to look like an unremarkable, cooperative email exchange right up until the moment the agent quietly writes to a file the defense was not built to watch [2].

This finding connects directly to a related attack class CSA's AI Safety Initiative documented days after MemGhost's disclosure. Agent Data Injection (ADI), described in research from Seoul National University, the University of Illinois Urbana-Champaign, and the security firm Largosoft, corrupts

metadata that an agent implicitly trusts — such as UI element identifiers, code authorship attribution, or fabricated tool-call results — rather than issuing direct instructions, and that research similarly found that guardrails purpose-built for prompt injection provided only partial coverage, with attack success rates against defended agents still reaching up to 50 percent [5]. MemGhost and ADI target different components of an agent's trust surface — durable memory state versus in-context metadata — but both demonstrate the same underlying architectural gap: agents that were hardened against instructions hidden in conversational input remain exposed when the attack instead targets the data the agent treats as already-verified fact.

### The Detection Problem

A recurring theme in the MemGhost paper is that the attack is not merely difficult to filter upstream; it is difficult for a human user to notice at all, by design. The agents under test hide their intermediate tool-use steps — file reads, file writes, memory updates — from the conversational transcript the user sees, a design choice made for readability rather than security, but one that MemGhost's payload is specifically engineered to exploit [1]. The researchers note plainly that "the assistant hides its behind-the-scenes steps by design, so the moment it edits a file never shows up in the chat," and that users rarely examine an agent's raw memory files even when they are stored in a human-readable format such as Markdown [1]. This is consistent with what CSA documented in its analysis of indirect prompt injection campaigns against OpenClaw, which found that identity and memory file poisoning of files such as SOUL.md and MEMORY.md can produce behavioral residue that persists even after a file is reverted, because vector-store indexes built from the poisoned content are not automatically purged alongside the underlying file [9]. The transport-layer defenses many organizations already rely on — SPF, DKIM, and DMARC email authentication — verify that a message genuinely originated from its claimed sender, but as the MemGhost researchers observe, authenticating where an email came from says nothing about whether its content should be allowed to become part of an agent's permanent, trusted state [2].

---

## Recommendations

### Immediate Actions

Organizations operating any AI agent with both persistent memory and autonomous email or messaging access should audit whether that agent's memory-write capability is available to background or unattended execution modes, since the research shows background processing to be substantially more exploitable than sessions where a user reviews the agent's reply [2]. As an immediate compensating

control, security teams should require that any inbound email or message processed by an agent be routed through a separate reader component that has no access to the agent's memory-write, file-write, or shell tools, passing only a bounded summary to the primary agent rather than the raw untrusted content — a mitigation consistent with general secure-agent-design guidance and directly aligned with the "lethal trifecta" pattern of private data access, untrusted content exposure, and external communication, a risk framing originally articulated by security researcher Simon Willison that CSA's agentic AI research has also highlighted as a recurring risk pattern in agent deployments [1]. Teams should also treat this disclosure as a prompt to inventory which of their deployed agents write to persistent memory files or vector stores at all, since the vulnerability is only exploitable where that capability exists.

## Short-Term Mitigations

Agent platforms and the organizations deploying them should implement provenance tagging that distinguishes memory entries derived from user statements from entries derived from external content such as email, so that externally sourced "facts" are flagged for confirmation rather than silently treated as equivalent to information the user explicitly provided [2]. Requiring explicit user confirmation before any write to a persistent memory file — rather than allowing the agent to update memory autonomously as a side effect of routine processing — would likely have blocked this specific attack scenario, at the cost of added friction for legitimate memory updates; organizations should weigh that tradeoff deliberately rather than defaulting to unattended writes for convenience. Audit logging of every memory modification, including the triggering context and source content, gives incident responders a forensic trail that the current generation of agents generally does not produce, and security teams should specifically request this capability from agent platform vendors during procurement or renewal conversations. Because the researchers found that input filters and safety-tuned models each missed the large majority of attempts, organizations should not treat vendor claims of prompt injection resistance as evidence of resistance to memory injection specifically; the two require independently validated defenses.

## Strategic Considerations

The MemGhost findings reinforce a broader governance gap that CSA's own enterprise survey research has already identified independent of this specific vulnerability. In a survey of 418 IT and security professionals, CSA found that 65 percent of organizations had experienced at least one AI agent-related security incident in the preceding twelve months [8], and that formal lifecycle and monitoring controls for autonomous agents remain immature relative to the pace of deployment, with only 21 percent of organizations reporting a formal decommissioning process for retired agents. Persistent memory

compounds this governance gap in a specific way: an agent's memory files are not a static configuration artifact reviewed once at deployment, but a continuously mutating store that the agent itself is permitted to write to as part of normal operation, which means the attack surface MemGhost exploits will exist for as long as an agent remains active, not merely at initial rollout. Security leaders evaluating or expanding personal AI agent deployments should treat memory-write auditability as a first-class requirement alongside more familiar controls such as tool-call authorization and network egress restriction, and should assume that any agent capable of autonomously updating its own long-term state is a candidate for the kind of stealthy, durable compromise this research demonstrates.

---

## CSA Resource Alignment

CSA's AI Safety Initiative published research on Agent Data Injection (ADI) days after MemGhost's disclosure, and the two findings are directly complementary: Agent Data Injection: A New Attack Class Beyond Prompt Injection documents that agents hardened against conventional prompt injection remain vulnerable when an attacker instead corrupts data the agent implicitly trusts, with defended agents still showing attack success rates up to 50 percent against metadata-level manipulation [5]. MemGhost demonstrates the same structural weakness applied to a different trust surface — persistent memory rather than in-context metadata — and the near-identical pattern of high-confidence defense bypass across both papers (memory-write filters missing MemGhost more than 90 percent of the time; prompt injection guardrails providing only partial coverage against ADI) suggests organizations should treat "trusted state corruption" as a single risk category spanning multiple technical vectors rather than as isolated, unrelated findings.

CSA's Agentic AI Threat Modeling Framework: MAESTRO provides the structural vocabulary for locating MemGhost within an organization's threat model. The attack spans MAESTRO's data operations and agent framework layers: the injection phase corrupts data at rest (the memory file itself), while the trigger phase exploits the agent framework's routine, trusted retrieval of that data during unrelated future sessions. Organizations conducting MAESTRO-based threat modeling of any agent with persistent memory should explicitly model memory-write and memory-read as distinct, independently securable operations rather than treating "memory" as a single undifferentiated component, since MemGhost's defense-evasion results show that controls addressing one do not automatically address the other.

CSA's AI Controls Matrix (AICM) v1.1 offers the control-mapping layer for organizations implementing the mitigations above. AICM's data security and identity and access management domains cover the provenance-tagging, confirmation-gating, and audit-logging controls this note recommends, and its tiered structure allows security teams to distinguish baseline expectations (any agent with memory-write

capability should log its writes) from more mature controls (memory writes require explicit, out-of-band user confirmation) as they build a maturity roadmap for agent deployments rather than attempting to implement every control simultaneously.

Finally, CSA's survey research, Autonomous but Not Controlled: AI Agent Incidents Now Common in Enterprises, supplies the organizational context for why vulnerabilities like MemGhost translate into real incidents rather than remaining theoretical: the same survey population reporting 65 percent incident prevalence also reported that visibility and lifecycle controls for autonomous agents lag adoption, which is precisely the condition under which a stealthy, silently-persisting compromise like MemGhost would go undetected for the longest period of time.

# References


[1] The Hacker News. "New MemGhost Attack Plants Persistent False Memories in AI Agents Through One Email." The Hacker News, July 2026.

[2] Zhang, Yechao; Zhao, Shiqian; Zhang, Jiawen; Zhang, Jie; Deng, Gelei; Liu, Xiaogeng; Xiao, Chaowei; Zhang, Tianwei. "When Claws Remember but Do Not Tell: Stealthy Memory Injection in Persistent Personal Agents." arXiv, July 6, 2026.

[3] The Hacker News. "ChatGPT macOS Flaw Could've Enabled Long-Term Spyware via Memory Function." The Hacker News, September 2024.

[4] Hack The Box. "Inside CVE-2025-32711 (EchoLeak): Prompt Injection Meets AI Exfiltration." Hack The Box, 2025.

[5] Cloud Security Alliance. "Agent Data Injection: A New Attack Class Beyond Prompt Injection." CSA AI Safety Initiative, July 16, 2026.

[6] Cloud Security Alliance. "Agentic AI Threat Modeling Framework: MAESTRO." Cloud Security Alliance, February 6, 2025.

[7] Cloud Security Alliance. "AI Controls Matrix (AICM) v1.1." Cloud Security Alliance, June 22, 2026.

[8] Cloud Security Alliance. "Autonomous but Not Controlled: AI Agent Incidents Now Common in Enterprises." Cloud Security Alliance, April 20, 2026.

[9] Cloud Security Alliance. "Trusted and Compromised: Indirect Prompt Injection in OpenClaw." CSA AI Safety Initiative, June 13, 2026.