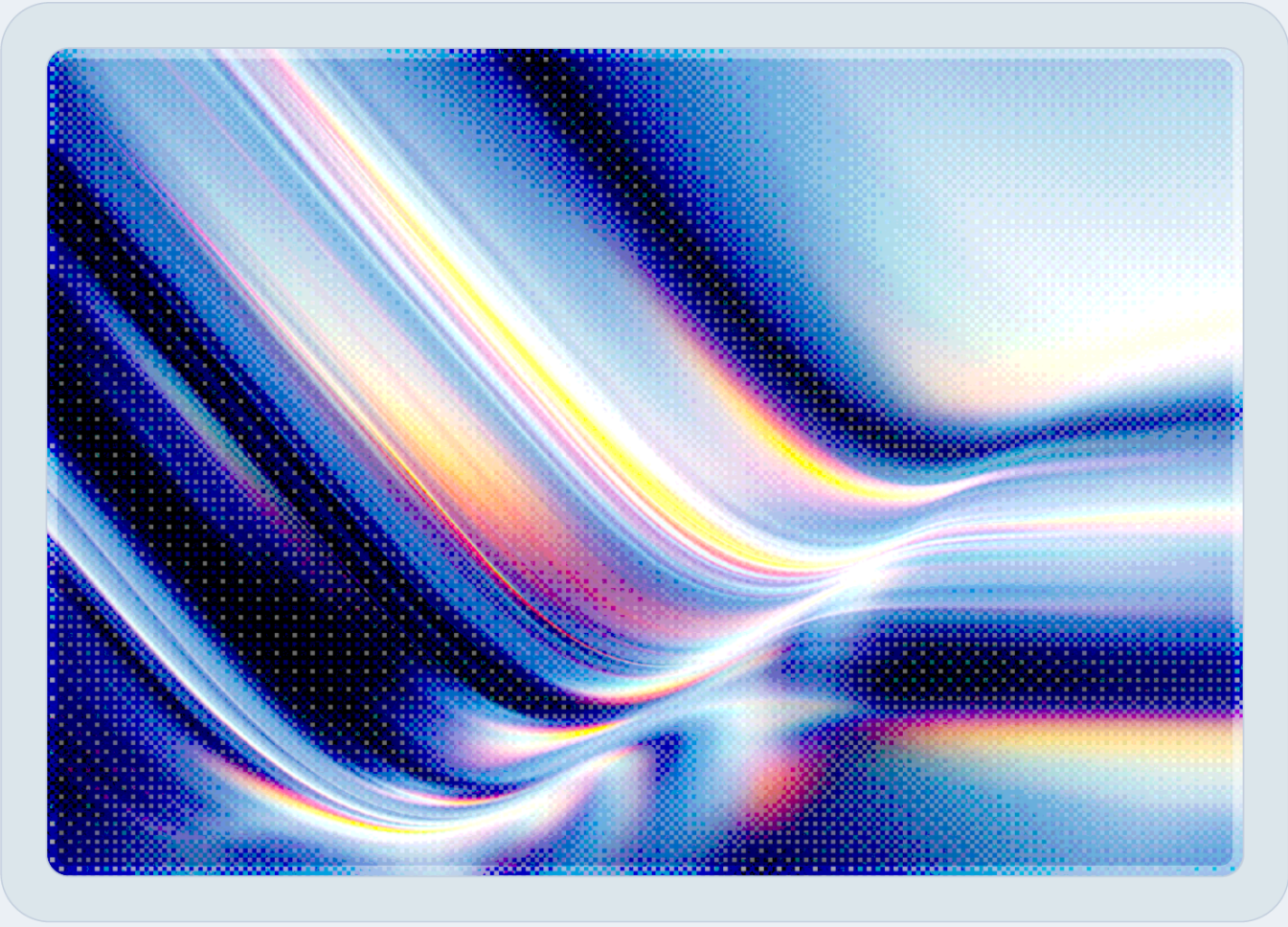
CSAI Foundation | Cloud Security Alliance

Claw Chain: TOCTOU Sandbox Escapes in AI Agent Runtimes

Security Implications of the OpenClaw Claw Chain Vulnerability Set

2026-05-16

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

- Cyera Research disclosed four chained vulnerabilities in OpenClaw, collectively named "Claw Chain," in May 2026. The chain enables data exfiltration, privilege escalation, and persistent backdoor implantation across an estimated 245,000 publicly accessible OpenClaw instances [1][3].
- Two of the four flaws are time-of-check/time-of-use (TOCTOU) race conditions in OpenClaw's managed sandbox backend (OpenShell), allowing attackers to redirect filesystem reads and writes outside the intended isolation boundary [2][4].
- A third vulnerability exploits a gap between command validation and shell execution to expand environment variables — including API keys and bearer tokens — through unquoted heredocs [1].
- The fourth vulnerability allows a local process with a valid bearer token to claim owner-level privileges by spoofing a client-controlled flag that OpenClaw did not validate against the authenticated session [1][2].
- All four flaws are patched in OpenClaw version 2026.4.22. Organizations running earlier versions should treat any unpatched OpenClaw deployment as potentially compromised until updated and audited.

Background

What OpenClaw Is

OpenClaw is a self-hosted, open-source autonomous AI agent framework built as a long-running Node.js service. It serves as a personal AI assistant that connects mainstream messaging channels — WhatsApp, Telegram, Discord, Slack, and dozens more — to large language model backends, then executes real-world tasks on the host system on the user's behalf [8]. Unlike stateless chatbots, OpenClaw maintains persistent context across conversations and can invoke more than one hundred preconfigured "AgentSkills" to execute shell commands, manage filesystems, automate browsers, and interact with external APIs.

The project reached 347,000 GitHub stars in April 2026, making it one of the most-starred repositories on the platform [8]. Search-engine indexing tools documented approximately 65,000 publicly reachable instances on Shodan and roughly 180,000 on ZoomEye, yielding a combined exposure surface of an estimated 245,000 servers [3]. For a self-hosted runtime that grants an AI agent privileged filesystem and shell access to the host machine, an internet-accessible exposure surface of this scale represents a substantial aggregate risk: each instance is a potential target for remote exploitation.

OpenShell and the Sandbox Model

The central element of the Claw Chain attack is OpenShell, OpenClaw's managed sandbox backend for code and command execution. OpenShell is designed to constrain agent activity to a defined filesystem mount root, preventing tool calls from escaping into the broader host environment. Sandbox isolation of this kind is among the most critical security mechanisms separating an AI agent's tool capabilities from sensitive host resources, because it operates at the layer where agent actions are directly executed. The Claw Chain research demonstrates that OpenShell's isolation was undermined by at least two independent race-condition flaws and two additional logic failures, any one of which could cause serious harm independently, and all four of which combine into a kill chain.

Disclosure and Patch Timeline

Cyera Research privately disclosed the four vulnerabilities to the OpenClaw maintainers in April 2026. An initial TOCTOU fix for the remote filesystem bridge's `readFile` function was released in version 2026.3.31 [4]. The full Claw Chain patch set — addressing all four advisories — was released on April 23, 2026 as version 2026.4.22 [1][2]. As of the date of this research note, all four vulnerabilities are patched and public advisories have been published under GHSA identifiers GHSA-5h3g-6xhh-rg6p, GHSA-wppj-c6mr-83jj, GHSA-r6xh-pqhr-v4xh, and GHSA-x3h8-jrgh-p8jx [1][5].

Security Analysis

The TOCTOU Vulnerability Class in AI Execution Contexts

Time-of-check/time-of-use (TOCTOU) vulnerabilities are a class of race condition defined under CWE-367 in which a security check and the subsequent use of the checked resource are not atomic operations. An attacker who can manipulate the resource between the check and the use — typically by

swapping a benign object for a malicious one — can bypass the check entirely [11]. In traditional systems, TOCTOU vulnerabilities most commonly appear in privileged filesystem operations: a process validates that a file path resolves safely, then a symlink is substituted before the actual I/O operation occurs.

In AI agent runtimes, TOCTOU vulnerabilities are more consequential than in most other software contexts for two reasons. First, the impacted component is typically the sandbox — the one control that separates the agent's tool execution from the broader host. Second, the agent itself may be the attacker's execution vehicle: a compromised or manipulated agent, whether through prompt injection or a malicious skill, can initiate the race from within the trust boundary without triggering conventional intrusion detection [2]. The Claw Chain findings illustrate both dynamics directly.

CVE-2026-44112: Sandbox Write Escape (CVSS 9.6)

The most severe of the four Claw Chain flaws, CVE-2026-44112, is a TOCTOU race condition in OpenShell's handling of filesystem write operations. The vulnerability arises because OpenClaw performs path validation and the subsequent write as two non-atomic steps [1][6]. An attacker who can win the race window between these two operations — for example, by swapping a validated path to a symlink pointing outside the mount root — can cause the agent to write arbitrary content to arbitrary locations on the host filesystem. The practical consequences of successful exploitation range from configuration tampering and credential file modification to backdoor implantation via the agent's own execution infrastructure. Because the write is performed with the agent's credentials and appears to originate from a normal tool call, it may not trigger alert conditions that would flag an unexpected process writing to sensitive locations. This vulnerability affects OpenClaw versions prior to 2026.4.22 [1].

CVE-2026-44113: Sandbox Read Escape (CVSS 7.7)

CVE-2026-44113 is the read-side counterpart to the write escape. The same structural deficiency — a non-atomic path-check-then-open sequence in OpenShell's remote filesystem bridge — allows an attacker to substitute a symlink in the race window between validation and file read [7][4]. The effect is that an agent's seemingly sandboxed read operation can be redirected to system files, credential stores, environment secrets, SSH keys, or any other material accessible to the agent's OS user. An early instance of this class, GHSA-9p3r-hh9g-5cmg, was patched in version 2026.3.31 [4]; the broader TOCTOU read exposure covering additional code paths was addressed in the April 23 full patch release [5].

CVE-2026-44115: Credential Exfiltration via Heredoc Expansion (CVSS 8.8)

The third vulnerability in the chain operates at the boundary between OpenClaw's command validation layer and its shell execution layer. When OpenClaw validates an incoming shell command, it inspects the command string before passing it to OpenShell for execution. However, for unquoted heredoc constructs, environment variable expansion occurs at the shell execution stage — after validation has already approved the command [1][12]. An attacker who can influence a command to include an unquoted heredoc referencing environment variables can cause sensitive runtime secrets to be returned through a command that appears benign at the validation stage. This vulnerability is particularly dangerous in the context of AI agent deployments where API keys, LLM provider tokens, and database credentials are routinely injected as environment variables, following standard container and runtime configuration patterns. The gap between the validation model and the execution model effectively creates a blind spot for an entire class of credential-harvesting payloads.

CVE-2026-44118: senderIsOwner Privilege Escalation (CVSS 7.8)

The fourth vulnerability is architectural rather than a race condition. OpenClaw's MCP loopback interface — the internal channel through which agents and skills communicate with the OpenClaw gateway — accepts a client-provided flag named `senderIsOwner` that signals whether the caller is authorized for owner-only tools and configuration operations [1][2][13]. The OpenClaw gateway trusted this flag without cross-referencing it against the authenticated session identity. A local process with a valid but non-owner bearer token could therefore present `senderIsOwner: true` and gain access to gateway configuration management, cron scheduling, and execution environment controls that should be restricted to the owner identity. This flaw transformed any process that could obtain a valid bearer token — itself facilitated by CVE-2026-44115 — into a de facto owner of the OpenClaw instance.

The Claw Chain Attack Sequence

What makes this vulnerability set particularly instructive for the broader AI security community is how the four flaws compose into a coherent kill chain. A threat actor who can influence a running agent's tool calls — through prompt injection into the agent's input channel, through a malicious AgentSkill, or through access to any connected messaging platform — can initiate the following sequence.

The chain begins with credential harvesting. The attacker induces the agent to execute a heredoc-bearing command that targets environment variables (CVE-2026-44115), or triggers a sandboxed file read that escapes the mount root via symlink substitution (CVE-2026-44113). Either path yields bearer tokens or API credentials held in the agent's runtime environment. With a valid token in hand, the attacker presents a spoofed `senderIsOwner` flag (CVE-2026-44118) to the MCP loopback,

elevating to owner-level control over the OpenClaw gateway. Finally, from that elevated position, the attacker exploits the write escape (CVE-2026-44112) to plant backdoors, modify cron schedules, or alter the agent's execution environment in ways that persist across restarts. Each step builds on the last; the full chain progresses from an agent manipulation primitive to durable host compromise without requiring any external network-level exploit [1][2].

Systemic Implications for AI Agent Security

The Claw Chain findings illustrate a recurring pattern in AI agent platform security: the attack surface created by giving AI agents privileged system capabilities is substantially larger than it first appears, and the existing security engineering techniques for constraining those capabilities may be underexamined in newly deployed frameworks.

TOCTOU flaws have been understood in operating system and filesystem security literature for decades. Their appearance in a newly dominant AI agent runtime — one that explicitly positions shell and filesystem access as core features — is consistent with a pattern in which newly deployed AI agent runtimes have not yet been subjected to the adversarial review applied to more mature sandboxed execution environments. When a framework achieves 347,000 GitHub stars and a quarter-million deployments, and a first published adversarial review of its sandbox reveals four chainable vulnerabilities, it raises questions about whether the security posture of AI agent frameworks is keeping pace with their adoption velocity [8][3]. Whether this represents a systemic gap across the AI agent framework ecosystem remains to be assessed by broader research.

The `senderIsOwner` vulnerability reflects a distinct failure mode: conflating authentication (does this caller have a token?) with authorization (is this caller permitted to claim owner-level access?). This confusion has been observed across a range of API systems where authorization semantics evolve after initial deployment, and the `senderIsOwner` pattern in OpenClaw is a concrete example of that broader vulnerability class. In AI agent runtimes that mediate real-world actions — filesystem modification, credential access, process scheduling — the consequences of that confusion are immediate and severe.

The heredoc expansion gap is a reminder that validation and execution models in any multi-stage pipeline can diverge in ways that create exploitable blind spots. As AI agent frameworks increasingly interpose validation layers (content filters, policy engines, human approval gates) between instruction receipt and action execution, each such boundary becomes a potential target for payload crafting that passes inspection but changes meaning at execution time. This pattern is closely related to prompt injection and deserves treatment as a first-class attack surface in agent runtime security design.

Recommendations

Immediate Actions

Organizations operating OpenClaw deployments should update to version 2026.4.22 or later without delay. Given that an estimated 245,000 instances were publicly reachable and the vulnerabilities were publicly disclosed in May 2026, unpatched instances should be treated as potentially compromised pending forensic review [3]. Operators should audit filesystem artifacts in locations writable by the agent user — particularly configuration files, cron entries, and shell initialization scripts — for unexpected modifications that could indicate exploitation of CVE-2026-44112 prior to patching.

Immediately following update, operators should rotate all secrets that were present in the agent's runtime environment variables: API keys, LLM provider tokens, database credentials, and OAuth tokens. CVE-2026-44115 may have permitted exfiltration of these secrets through interactions that left no obvious trace in application logs.

Short-Term Mitigations

For organizations that cannot immediately update, reducing the OpenClaw instance's network exposure provides meaningful risk reduction. Instances should not be reachable from the public internet unless that exposure is intentional and operationally required. Firewall rules, VPN requirements, or reverse-proxy authentication layers in front of the OpenClaw gateway substantially limit the population of potential attackers who can interact with the MCP loopback interface or manipulate agent inputs.

Operators should review which messaging channels are connected to their OpenClaw instance and audit the trust model for each. Channels that accept messages from untrusted members of the public — public Discord servers, open WhatsApp groups, publicly accessible webhooks — are the highest-risk input vectors for prompt injection attacks that could initiate the Claw Chain sequence. Restricting inbound message sources to trusted identities substantially reduces the population of actors who could initiate the Claw Chain sequence through agent input manipulation, though it does not eliminate prompt injection risks from other input surfaces such as retrieved web content or file inputs.

Environment variable hygiene deserves immediate attention regardless of patch status. Secrets that do not need to be present in the agent's runtime environment should be removed. Where secrets must be present, consider whether they can be scoped to narrower permissions — a read-only database credential instead of a read-write one, a token with limited API access rather than an administrative one — so that exfiltration via CVE-2026-44115 yields lower-value material.

Strategic Considerations

The Claw Chain research should prompt organizations to broaden how they evaluate AI agent frameworks during procurement and deployment. The capabilities that make an AI agent powerful — shell access, filesystem access, credential use, long-running process scheduling — are exactly the capabilities that create the most severe attack surface when sandbox isolation fails. Security teams should treat AI agent runtime isolation as a critical security control requiring the same adversarial testing applied to browser sandboxes, container runtimes, and hypervisors: not assumed correct by design, but verified through structured red-team assessment.

Sandbox isolation testing for AI agent runtimes should explicitly include TOCTOU testing in filesystem bridge implementations. The attack pattern is well understood, the exploitation techniques are documented, and the tooling to test for symlink race conditions in filesystem operations exists in open-source form. Any managed execution sandbox that does not atomize its path-validation and file-operation steps into a single kernel-level operation should be treated as potentially vulnerable to this class of attack.

The `senderIsOwner` flaw argues for requiring AI agent frameworks to implement explicit authorization models rather than relying on client-provided claims. Trust elevation should require a verifiable property of the authenticated session — such as the identity under which the session was established — rather than a field the caller can set unilaterally. Operators evaluating AI agent platforms should ask vendors and maintainers directly how privilege levels are determined and whether any privilege claim depends on client-controlled data that is not independently verified.

CSA Resource Alignment

MAESTRO

CSA's MAESTRO framework for agentic AI threat modeling is a directly applicable analytical lens for the Claw Chain vulnerabilities, particularly given its explicit Layer 3 treatment of agent framework execution environments [9]. The OpenShell TOCTOU flaws map to Layer 3 of the MAESTRO architecture — the Agent Frameworks layer — which encompasses the reasoning loop, tool dispatch, and the execution environment in which agent actions are carried out. MAESTRO explicitly catalogs sandbox isolation failure as a Layer 3 threat, with downstream consequences that propagate into Layer 2 (Data and Context) when file reads escape the sandbox boundary and expose credentials. The senderIsOwner

escalation maps to MAESTRO's treatment of agent impersonation and authorization bypass threats at Layer 3, where the framework identifies insufficient identity verification within the agent runtime as a high-priority risk vector.

The Claw Chain attack sequence — from agent input manipulation through credential harvest, privilege escalation, and persistence — also exemplifies the cross-layer lateral movement scenarios that MAESTRO documents as compound threats. Security teams using MAESTRO to model their agentic AI deployments should explicitly include the question of whether their agent framework's sandbox backend uses atomic path operations, and whether any privilege mechanism accepts client-controlled trust claims without session-level verification.

AI Controls Matrix (AICM)

CSA's AI Controls Matrix (AICM) addresses execution environment security under its Orchestrated Service Provider (OSP) domain, which covers the controls applicable to platforms that manage agent execution on behalf of users or organizations [10]. The Claw Chain findings are directly relevant to AICM controls covering sandbox isolation, least-privilege execution, and credential management within AI execution environments. The AICM's shared security responsibility model highlights that when an OSP such as OpenClaw provides a managed execution environment, the platform provider bears responsibility for the integrity of the isolation boundary — a responsibility that the Claw Chain vulnerabilities demonstrate was not adequately fulfilled in pre-patch versions of OpenShell.

Operators mapping their OpenClaw deployments against the AICM should treat the pre-patch OpenClaw versions as failing the relevant sandbox isolation controls, and should document remediation steps — patching, secret rotation, network restriction, and forensic review — as evidence of control restoration.

Zero Trust Considerations

The Claw Chain attack chain is structurally compatible with Zero Trust principles as a countermeasure framework. A Zero Trust posture requires that no internal claim — including a flag set by a local process — be trusted without independent verification. The `senderIsOwner` flaw is precisely the scenario Zero Trust architecture exists to prevent: an internal caller that presents a self-asserted privilege claim that is accepted without cryptographic or session-level verification. Organizations applying Zero Trust principles to their AI agent infrastructure would be more likely to surface the `senderIsOwner` pattern as a design deficiency during architecture review, because Zero Trust architecture explicitly requires that privilege elevation depend on verifiable session properties rather than self-asserted caller claims.

References

[1] Cyera Research. "Claw Chain: Cyera Research Unveil Four Chainable Vulnerabilities in OpenClaw." Cyera, May 2026.

[2] The Hacker News. "Four OpenClaw Flaws Enable Data Theft, Privilege Escalation, and Persistence." The Hacker News, May 2026.

[3] Cybersecurity News. "OpenClaw Chain Vulnerabilities Expose 245,000 Public AI Agent Servers to Attack." Cybersecurity News, May 2026.

[4] GitHub Security Advisory. "Sandbox escape via TOCTOU race in remote FS bridge readFile (GHSA-9p3r-hh9g-5cmg)." GitHub, April 2026.

[5] GitHub Security Advisory. "OpenShell FS bridge reads pin and verify the opened file before returning bytes (GHSA-5h3g-6xhh-rg6p)." GitHub, April 2026.

[6] ThreatINT. "CVE-2026-44112." ThreatINT, 2026.

[7] ThreatINT. "CVE-2026-44113." ThreatINT, 2026.

[8] GitHub. "openclaw/openclaw — Your own personal AI assistant." GitHub, 2026.

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

[10] Cloud Security Alliance. "AI Controls Matrix (AICM)." CSA, 2025.

[11] DeepStrike. "What Is Time of Check Time of Use (TOCTOU)? Explained." DeepStrike, 2026.

[12] ThreatINT. "CVE-2026-44115." ThreatINT, 2026.

[13] ThreatINT. "CVE-2026-44118." ThreatINT, 2026.