ClawJacked: WebSocket Exploitation Enabling Malicious Sites to Hijack Local AI Agents

Unofficial AI-assisted Research

Cloud Security Alliance AI Safety Initiative

2026-03-10

Key Takeaways

This research note documents ClawJacked, a compound vulnerability chain in the OpenClaw AI agent framework disclosed on February 25, 2026, and situates it within a broader wave of parallel disclosures affecting the Model Context Protocol (MCP) ecosystem. Together, these disclosures reveal a structural security deficit in local AI agent tooling that patch cycles alone will not resolve. The following points summarize the core findings.

- ClawJacked, disclosed by Oasis Security on February 25, 2026, is a vulnerability chain in the OpenClaw AI agent framework that allows any malicious website to silently seize full, authenticated control of a locally running OpenClaw instance — requiring no plugins, extensions, or active user interaction beyond visiting a page [1][2].
- The attack exploits three compounding design oversights in OpenClaw's local WebSocket gateway: the browser's same-origin policy does not prevent web pages from opening WebSocket connections to `localhost`, rate limiting was entirely disabled for loopback-origin connections, and device pairing was auto-approved for any client presenting from the local interface [1][3].
- ClawJacked is among the highest-profile instances of a broader structural vulnerability class that has simultaneously affected the Anthropic MCP Inspector (CVE-2025-49596, CVSS 9.4), the official MCP Python and TypeScript SDKs (CVE-2025-66416, CVE-2025-66414), Microsoft's Playwright MCP Server (CVE-2025-9611), and SillyTavern (CVE-2025-59159, CVSS 9.7), among others [4][5][6][7].
- OpenClaw shipped a fix in v2026.2.25 within 24 hours of the February 25, 2026 disclosure; organizations and developers running locally hosted AI agent infrastructure should treat updates to OpenClaw and to the MCP SDK ecosystem as urgent patch actions [1][8].
- The root cause pattern — local AI servers listening on WebSocket or HTTP endpoints without Host header validation, origin enforcement, or authentication — is widespread across the tooling ecosystem and will not be resolved by point patches alone; architectural controls at the framework and deployment level are required [4][9].

Background

The proliferation of locally running AI agent infrastructure has introduced a new attack surface that the security community is only beginning to systematically characterize. Developer workstations in AI-forward organizations commonly run OpenClaw instances, Model Context Protocol (MCP) servers, AI coding assistants, and similar tools as persistent local services that communicate via WebSocket or HTTP on loopback ports. These services operate under the implicit assumption that code executing on the same machine is trusted — an assumption inherited from a pre-agentic era when local services primarily served processes spawned by the same user. That assumption is incorrect in the modern threat environment, because the browser, which executes arbitrary web content from untrusted origins, also runs on the same machine and can reach any locally bound port that the operating system's firewall does not explicitly restrict.

Browser security architecture does not solve this problem. The same-origin policy, which prevents web pages from making arbitrary cross-origin HTTP requests, applies only to fetch and XMLHttpRequest calls; it does not block WebSocket connections from a web page's JavaScript to a `ws://localhost:<port>` endpoint. DNS rebinding techniques further extend this exposure: an attacker who controls a DNS record can cause a browser to reclassify a connection to an attacker-controlled server as a localhost connection, bypassing even the limited origin controls that some local services implement by checking the `Host` header. These are not theoretical attack paths — they are the mechanisms underlying ClawJacked and the wave of parallel disclosures that accompanied it.

OpenClaw is an AI agent framework used by developers to orchestrate local and cloud-connected AI workflows. Its local gateway component listens on a WebSocket port to coordinate agent tasks, device pairing, and configuration management. Prior to February 26, 2026, that gateway contained a combination of design choices that made it exploitable by any web page a developer happened to visit. Oasis Security researchers identified and responsibly disclosed the compound vulnerability chain on February 25, 2026; OpenClaw released v2026.2.25 incorporating a fix within approximately 24 hours [1] [2]. The rapid patch turnaround suggests OpenClaw had prepared a fix during the coordinated disclosure process, though it also underscores the severity of the risk. Enterprise and developer adoption of the update depends on awareness that the patch is both available and necessary.

Security Analysis

The ClawJacked Vulnerability Chain

ClawJacked is not a single flaw but a chain of three compounding design oversights that, in combination, enable full agent takeover from a browser tab. Understanding each layer individually is important because fixing any one layer in isolation would not have prevented exploitation, and because analogous combinations may exist in other local AI agent systems that share similar design patterns.

The first layer is the absence of WebSocket origin enforcement. When a web page's JavaScript opens a WebSocket connection to `ws://localhost:<port>`, the browser transmits an `Origin` header indicating the requesting page's domain. OpenClaw's gateway did not validate this header against an allowlist of trusted origins, meaning connections originating from arbitrary websites were accepted as if they were legitimate local clients. This is the foundational permissiveness that gives a malicious page its initial channel into the gateway [1][3].

The second layer is the loopback exemption in rate limiting. OpenClaw implemented request throttling to protect against brute-force authentication attacks, but explicitly disabled that throttling for connections arriving from the loopback interface. The exemption was apparently designed to avoid throttling legitimate local tooling, but it assumes the loopback interface itself is trusted. Browser JavaScript connecting to localhost via WebSocket arrives, from the gateway's perspective, as a loopback-origin connection, making the brute-force protections completely inoperative against attacker-controlled browser code. Oasis Security researchers reported that this allowed hundreds of authentication guesses per second — sufficient, at that rate, to exhaust a typical common-password list within seconds to minutes [3].

The third layer is automatic device pairing from localhost. OpenClaw's device authorization flow — the mechanism through which new clients are granted persistent, authenticated access — automatically approved pairing requests originating from the local interface without presenting a confirmation prompt to the user. Once a malicious web page's WebSocket connection successfully authenticated through the brute-forced gateway, it could register itself as a trusted device silently, establishing persistent access that survives browser navigation away from the malicious page [1][3].

The aggregate effect is that a developer visiting a malicious web page — embedded in a phishing email, a malicious advertisement, or a typosquat of a developer resource — could have their entire OpenClaw instance taken over without any visible indication. Post-compromise, an attacker with full gateway access can issue arbitrary instructions to the AI agent, enumerate and interact with paired devices, read agent logs and configuration, access the local filesystem through agent-mediated operations, and execute

arbitrary shell commands on the workstation through agent tool use. The blast radius extends to any data or system the AI agent has been configured to access, which in developer environments frequently includes source code repositories, environment files containing cloud credentials, internal API endpoints, and production access credentials [1][2].

CVE-2026-25253 and the Log Poisoning Companion Vulnerability

ClawJacked exists in context alongside two earlier OpenClaw vulnerabilities that share its underlying trust model problems, together comprising a three-wave disclosure sequence against the same product. CVE-2026-25253, patched silently in OpenClaw v2026.1.29 on January 29, 2026, and disclosed publicly by SecurityWeek on February 3, 2026, involved a different but related WebSocket exploitation path: versions prior to 2026.1.29 accepted a `gatewayUrl` query parameter that caused OpenClaw to automatically establish a WebSocket connection to an attacker-specified URL and transmit the user's authentication credentials in the process [8][10]. Crafting a malicious link that would trigger this behavior required no technical access to the victim's machine — only that the victim click a link. SOCRadar researchers characterized this as a one-click remote code execution path because the exfiltrated authentication token provided full gateway access through which RCE was achievable via agent tool invocation [10].

A third vulnerability, tracked as GHSA-g27f-9qjv-22pm, addressed the risk of indirect prompt injection through OpenClaw's log subsystem in versions prior to v2026.2.13 [11]. The gateway logged WebSocket request headers verbatim, without sanitization or length limits, to an endpoint accessible on TCP port 18789. Because OpenClaw agents periodically review their own logs as part of troubleshooting workflows, an unauthenticated attacker could submit a WebSocket connection whose headers contained crafted LLM instructions; those instructions would be written to the log and subsequently processed as agent directives when the log endpoint was consulted. This indirect prompt injection path required no authentication and no prior compromise of the gateway — only network access to port 18789, which on developer workstations is typically unfiltered for local network traffic [11].

Taken together, these three disclosures reveal an OpenClaw codebase that did not apply consistent security discipline to its local WebSocket and HTTP interfaces. Each individual vulnerability is explicable as an oversight; their occurrence in sequence within a six-week window may suggest that a systematic review of the gateway trust model was not initiated until external researchers applied scrutiny, though it is also possible that the first disclosure triggered an internal audit that surfaced the subsequent issues. The sequence carries meaningful signal for vendor risk evaluation regardless of which interpretation is more accurate.

CVE / Advisory	Severity	Issue	Patched Version	Disclosure Date
CVE-2026-25253	CVSS 8.8	Auth token exfiltration via `gatewayUrl` parameter leading to RCE	v2026.1.29	Feb 3, 2026 [8][10]
GHSA-g27f-9qjv-22pm	Low (CVSS 3.1)	Log poisoning / indirect prompt injection via WebSocket headers on port 18789	v2026.2.13	Feb 13, 2026 [11]
ClawJacked	High	WebSocket brute-force + auto device registration from localhost	v2026.2.25	Feb 25–26, 2026 [1][3]

The Broader MCP and Local Agent Ecosystem

ClawJacked is among the highest-profile vulnerabilities in a wave of parallel disclosures that collectively document the same structural problem across the AI agent tooling ecosystem. The pattern is consistent: local AI infrastructure components listen on HTTP or WebSocket endpoints under the assumption that only trusted local code will connect to them, and that assumption is violated by browser JavaScript executing attacker-controlled content in the same network namespace.

Oligo Security's disclosure of CVE-2025-49596 (CVSS 9.4) against Anthropic's own MCP Inspector development tool demonstrates that even the AI framework vendor's reference tooling was susceptible [4]. The MCP Inspector, which developers use to test and debug MCP server implementations, exposed a localhost WebSocket endpoint without DNS rebinding protection. A developer visiting a malicious page while running MCP Inspector could have the tool's full debugging capabilities — which include reading server configuration, sending arbitrary MCP protocol messages, and interacting with filesystem and shell tools on any connected MCP server — silently commandeered [4]. Anthropic and Docker jointly disclosed the vulnerability; Docker's subsequent write-up noted that the scenario illustrates how a local development tool can serve as a pivot point into a developer's entire MCP server configuration [12].

The official MCP SDK implementations for Python and TypeScript shipped without DNS rebinding protection enabled by default, documented respectively as CVE-2025-66416 and CVE-2025-66414 [5] [6]. These vulnerabilities affect any application built on the official SDKs that exposes an HTTP-based MCP transport endpoint on localhost without explicitly enabling host validation — a default configuration

that many developers would not recognize as insecure. The MCP Python SDK addressed this in version 1.23.0 by enabling automatic DNS rebinding protection when the server is bound to `127.0.0.1` or `localhost` [5]. The TypeScript SDK received a parallel fix. Given the large number of MCP server implementations in the ecosystem, however, organizations should treat the SDK fixes as necessary but not sufficient: custom and third-party MCP servers that were built against older SDK versions may require independent remediation.

Additional disclosures in the same vulnerability class include CVE-2025-9611 against Microsoft's Playwright MCP Server (DNS rebinding via missing Origin header validation), CVE-2025-59956 against the Coder Agent API (DNS rebinding exposing Claude Code message history including secrets), and CVE-2025-59159 (CVSS 9.7) against SillyTavern (full remote control of local AI instances via DNS rebinding) [7][9][13]. The breadth of affected products confirms that this is not a vendor-specific failure but a category-level problem rooted in the absence of shared secure defaults across the tooling ecosystem.

Developer Workstation as High-Value Target

The organizational risk calculus for these vulnerabilities is different from that of conventional enterprise application vulnerabilities, where the affected systems are well-defined servers in managed infrastructure. Local AI agent tools run on developer, researcher, and data science workstations — endpoints that are simultaneously high-privilege (developers frequently hold administrative access to production environments, source code, cloud accounts, and internal services) and, in many organizations, under-monitored relative to server infrastructure, where developer workflow traffic can be difficult to baseline. A threat actor who successfully executes ClawJacked or an analogous exploit against a developer's workstation receives, through the AI agent, a capability-rich pivot point into everything that agent has access to.

The attack delivery mechanism — a web page that opens a WebSocket connection to localhost — is compatible with a wide range of delivery channels. Spear phishing emails targeting developers, malvertising on developer-oriented platforms, typosquatting of npm package documentation pages, and compromise of legitimate developer resource websites are all plausible vehicles. The attack requires no malware installation, no privilege escalation on the host OS, and is unlikely to trigger conventional host-based detections that focus on malicious process execution; however, EDR platforms with browser inter-process communication monitoring may surface the anomalous WebSocket communication depending on tuning and configuration. The WebSocket connection from the browser to localhost is otherwise indistinguishable at the network layer from a legitimate local tool interaction [1][3].

Recommendations

Immediate Actions

Organizations and individual developers running OpenClaw should update to v2026.2.25 or later immediately. Environments that have not yet applied the earlier patches should also confirm they are running at minimum v2026.1.29 (CVE-2026-25253 fix) and v2026.2.13 (log poisoning fix) as stepping stones. Update status should be verified through OpenClaw's version reporting rather than assumed based on installation date, as silent patching in v2026.1.29 means some administrators may be unaware that intermediate updates were released.

Developers building on or deploying MCP server infrastructure should update the MCP Python SDK to version 1.23.0 or later and apply the TypeScript SDK patch for CVE-2025-66414. For any MCP server that was built against SDK versions predating these fixes, the server-side code should be reviewed to confirm that Host header validation and origin checks are enforced. The MCP Inspector development tool should be updated to the patched version addressing CVE-2025-49596, and developers should treat the Inspector as a sensitive tool that is only opened when actively needed rather than run as a persistent background service.

Security operations teams should assess whether developer workstations in the environment run locally bound AI agent services and confirm that those services are not listening on all interfaces (`0.0.0.0`) rather than exclusively on localhost. Services that bind to all interfaces are reachable from the local network and should be reconfigured; services bound only to loopback are exposed exclusively to same-machine processes, including browser JavaScript, against which network-layer firewall rules provide no defense. The primary ClawJacked threat originates from browser JavaScript executing on the same machine, not from external network sources.

Short-Term Mitigations

At the application layer, any local AI service that accepts WebSocket or HTTP connections should implement three controls: validation of the `Host` header against an explicit allowlist, validation of the `Origin` header for WebSocket upgrade requests, and rate limiting that applies uniformly to connections from all source addresses including the loopback interface. Authentication flows for device pairing or session establishment should require explicit user confirmation regardless of connection source address. These controls collectively close the specific attack surface exploited by ClawJacked and its analogues.

Organizations that cannot immediately update affected tooling should consider browser-level mitigations as a compensating control. Configuring developer browser profiles to use enterprise DNS resolvers that do not resolve attacker-controlled domains to localhost addresses partially mitigates DNS rebinding attacks, though it does not address direct `ws://localhost` connections of the type used in ClawJacked. Browser security extensions that explicitly block JavaScript from initiating WebSocket connections to loopback addresses represent a more targeted mitigation but must be evaluated for compatibility with legitimate developer tooling that communicates via localhost WebSocket.

Threat detection coverage should be updated to alert on WebSocket connection attempts from browser processes to loopback-bound ports associated with known AI agent tools. Host-based detection rules that identify when a browser child process communicates to OpenClaw's gateway port, or when the OpenClaw process makes unexpected outbound connections following a period of browser activity, may surface exploitation attempts that network-layer controls miss.

Strategic Considerations

The vulnerability class documented here — local AI services exposed to browser-origin WebSocket connections — will not be remediated through patch cycles applied to existing products alone. The ecosystem contains a rapidly growing number of MCP servers, AI agent runtimes, and developer tools built against insecure defaults, and the population is expanding rapidly, placing pressure on the security review processes that must keep pace [5][9]. Sustainable improvement requires that the MCP protocol specification, AI agent framework SDKs, and tooling scaffolding enforce secure-by-default configurations: Host header validation and origin enforcement should be enabled without developer action, brute-force protection should apply unconditionally, and device pairing flows should require explicit user confirmation as a non-optional UX element.

Security teams at organizations that develop or operate AI agent tooling should incorporate localhost WebSocket security into their application security review checklists, threat modeling exercises, and red team scope definitions. CSA's MAESTRO threat modeling framework is particularly applicable for characterizing the trust boundary between browser-executed content and locally running agent infrastructure: its Layer 4 (Agent Runtime) analysis should explicitly address inbound connection surfaces and the conditions under which those connections are trusted. Organizations that do not apply this scrutiny should expect to encounter ClawJacked-class vulnerabilities in their own tooling or in third-party AI tools they adopt.

CSA Resource Alignment

MAESTRO, the CSA agentic AI threat modeling framework, is the most directly applicable CSA resource for addressing the vulnerability class documented in this note. MAESTRO's hierarchical risk model addresses agent trust boundaries and the conditions under which external inputs — including inputs arriving via network interfaces — are granted authority to direct agent behavior [14]. The implicit trust that OpenClaw and analogous tools extended to loopback-origin connections is precisely the type of assumption that MAESTRO's Layer 4 (Agent Runtime) and Layer 6 (Orchestration) analyses are designed to surface and challenge. Organizations deploying AI agent infrastructure should complete MAESTRO-based threat models that explicitly enumerate inbound connection surfaces, classify the trust level of each connection source, and verify that authentication and authorization controls are not predicated on network-layer assumptions that browser execution can violate.

The CSA AI Controls Matrix (AICM) v1.0 provides control coverage applicable to several dimensions of this threat. Domain areas covering AI supply chain security apply to the SDK-level vulnerabilities in the MCP Python and TypeScript SDKs, where insecure defaults in the framework propagated risk to all applications built on it — a supply chain trust issue as much as an individual product flaw. Shared security responsibility domain controls apply to the question of where responsibility for localhost service security lies between AI framework vendors, tool developers who build on those frameworks, and organizations that deploy the resulting tools in developer workstation environments. Agentic AI governance controls address the need for organizations to maintain inventories of locally running AI services and apply patch management processes to them with the same urgency as internet-facing applications.

The CSA Cloud Controls Matrix (CCM) threat and vulnerability management domain (TVM) is relevant to the patch cadence dimension of this incident. The sequence of three OpenClaw patches within six weeks, combined with parallel disclosures across the MCP ecosystem, represents a period of sustained vulnerability activity that demands aggressive patch SLA performance. CCM TVM controls define patch timeframes based on severity; organizations should review whether their policies apply those SLAs to developer workstation tooling as well as server infrastructure, and should audit OpenClaw and MCP SDK version distribution across managed endpoints.

Zero Trust principles apply directly to the design failures documented here. The core Zero Trust imperative — never trust, always verify — is violated by any local service that treats connection source address as a trust signal. Localhost is not a trust boundary in an environment where browser JavaScript executes alongside local services; applying Zero Trust architecture to AI agent runtime design requires that authentication, authorization, and origin verification be enforced for every connection regardless of

whether it arrives from the loopback interface. Organizations that have implemented Zero Trust for their cloud and network infrastructure should extend that framework's principles explicitly to local AI agent services as part of their developer workstation security program.

CSA's STAR program and AI organizational responsibilities guidance are relevant for organizations evaluating AI agent tool vendors. The ClawJacked sequence — three significant vulnerabilities disclosed against a single product within six weeks, with one patched silently before public disclosure — provides meaningful signal about a vendor's approach to security engineering. Procurement and vendor risk assessment processes should include questions about secure development lifecycle practices, localhost service security architecture, and vulnerability disclosure and patching policies as standard due diligence for AI tooling.

References

[1] S. Lakshmanan, "ClawJacked Flaw Lets Malicious Sites Hijack Local OpenClaw AI Agents via WebSocket," The Hacker News, February 2026. https://thehackernews.com/2026/02/clawjacked-flaw-lets-malicious-sites.html

[2] BleepingComputer, "ClawJacked attack let malicious websites hijack OpenClaw to steal data," BleepingComputer, February–March 2026. https://www.bleepingcomputer.com/news/security/clawjacked-attack-let-malicious-websites-hijack-openclaw-to-steal-data/

[3] Oasis Security, "ClawJacked: OpenClaw Vulnerability Enables Full Agent Takeover," Oasis Security Blog, February 25, 2026. https://www.oasis.security/blog/openclaw-vulnerability

[4] Oligo Security, "Critical RCE Vulnerability in Anthropic MCP Inspector (CVE-2025-49596)," Oligo Security Blog, 2025. https://www.oligo.security/blog/critical-rce-vulnerability-in-anthropic-mcp-inspector-cve-2025-49596

[5] GitHub Advisory Database, "CVE-2025-66416: MCP Python SDK — No DNS Rebinding Protection by Default," GitHub, 2025. https://github.com/advisories/GHSA-9h52-p55h-vw2f

[6] GitLab Advisory Database, "CVE-2025-66414: MCP TypeScript SDK — No DNS Rebinding Protection by Default," GitLab, 2025. https://advisories.gitlab.com/pkg/npm/@modelcontextprotocol/sdk/CVE-2025-66414/

[7] Security Online, "Critical Flaw CVE-2025-59159 (CVSS 9.7) in SillyTavern Allows Full Remote Control of Local AI Instances," Security Online, 2025. https://securityonline.info/critical-flaw-cve-2025-59159-cvss-9-7-in-sillytavern-allows-full-remote-control-of-local-ai-instances/

[8] SecurityWeek, "OpenClaw Vulnerability Allowed Websites to Hijack AI Agents," SecurityWeek, February 3, 2026. https://www.securityweek.com/openclaw-vulnerability-allowed-malicious-websites-to-hijack-ai-agents/

[9] MCP Security Research, "Coder's Agent API Exposes User Chat History Via DNS Rebinding (CVE-2025-59956)," mcpsec.dev, September 2025. https://mcpsec.dev/advisories/2025-09-19-coder-chat-exfiltration/

[10] SOCRadar, "CVE-2026-25253: 1-Click RCE in OpenClaw via Auth Token Exfiltration," SOCRadar Blog, February 2026. https://socradar.io/blog/cve-2026-25253-rce-openclaw-auth-token/

[11] GitLab Advisory Database, "GHSA-g27f-9qjv-22pm: OpenClaw Log Poisoning / Indirect Prompt Injection via WebSocket Headers," GitLab, February 2026. https://advisories.gitlab.com/pkg/npm/openclaw/GHSA-g27f-9qjv-22pm/

[12] Docker, "MCP Horror Stories: The Drive-By Localhost Breach (CVE-2025-49596)," Docker Blog, 2025. https://www.docker.com/blog/mpc-horror-stories-cve-2025-49596-local-host-breach/

[13] Dark Reading, "Critical Vulnerability in OpenClaw: AI Agent Risks in Local Environments," Dark Reading, February–March 2026. https://www.darkreading.com/application-security/critical-openclaw-vulnerability-ai-agent-risks

[14] Cloud Security Alliance, "MAESTRO: Agentic AI Threat Modeling Framework," CSA, 2025. https://cloudsecurityalliance.org/

Further Reading

The following sources provided additional background and corroborating coverage for the ClawJacked vulnerability and the broader DNS rebinding risk class affecting local AI agent infrastructure. They are included here for readers who wish to explore the topic further.

[15] OECD.AI Incident Database, "Critical OpenClaw AI Vulnerability Allows Malicious Websites to Hijack Local AI Agents," OECD.AI, March 1, 2026. https://oecd.ai/en/incidents/2026-03-01-476b

[16] Privacy Guides, "ClawJacked Vulnerability Allows Malicious Websites to Take Control of OpenClaw," Privacy Guides, March 2, 2026. https://www.privacyguides.org/news/2026/03/02/clawjacked-vulnerability-allows-malicious-websites-to-take-control-of-openclaw/

[17] Straiker, "Agentic Danger: DNS Rebinding Exposing Your Internal MCP Servers," Straiker Blog, 2025. https://www.straiker.ai/blog/agentic-danger-dns-rebinding-exposing-your-internal-mcp-servers

[18] Security Affairs, "ClawJacked flaw exposed OpenClaw users to data theft," Security Affairs, March 2026. https://securityaffairs.com/188749/hacking/clawjacked-flaw-exposed-openclaw-users-to-data-theft.html