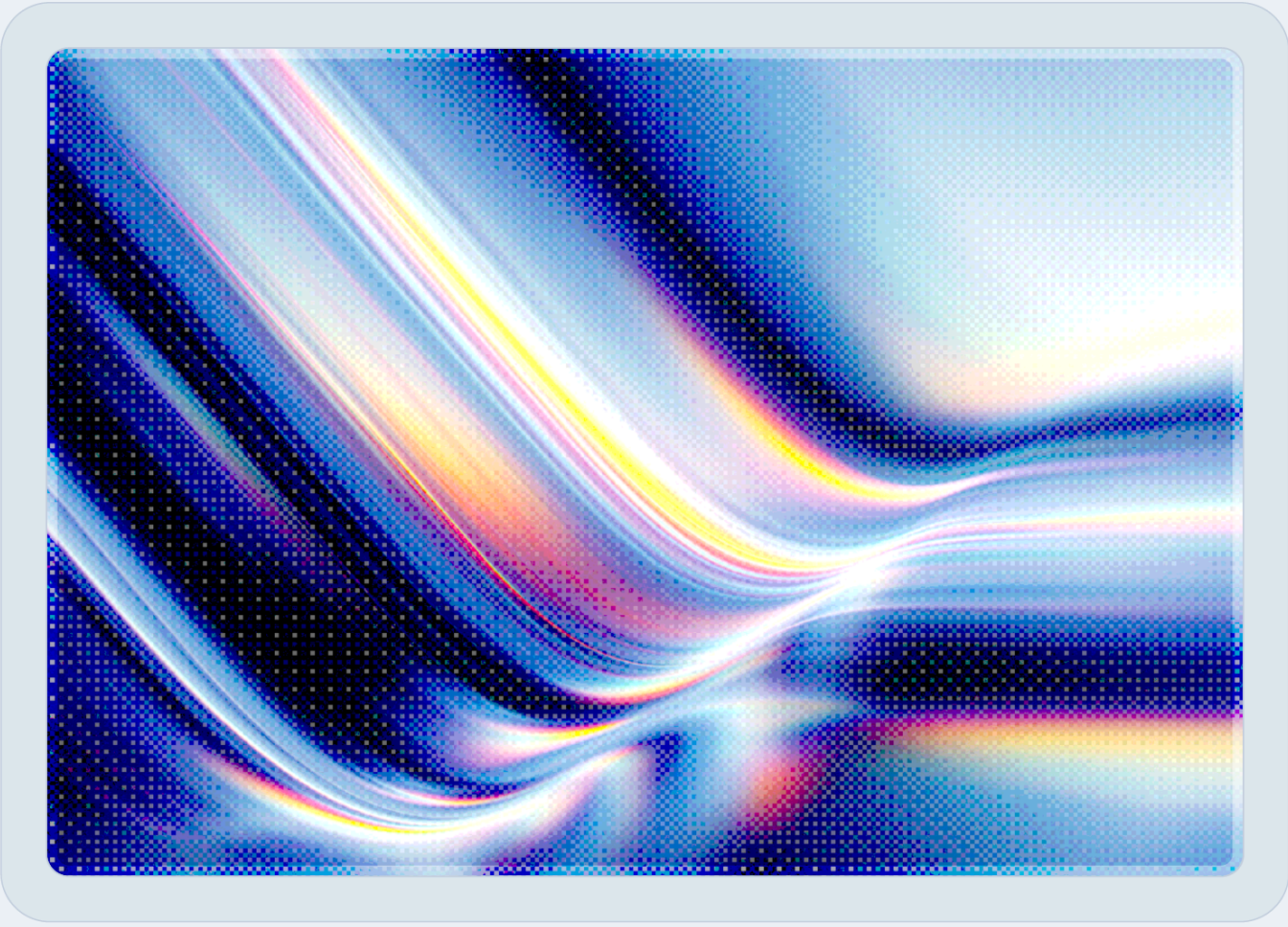
CSAI Foundation | Cloud Security Alliance

BragJack: One Extension Breaks Five Browsers' AI Trust Model

A Shared Architectural Flaw Across Chrome, Comet, Edge, Opera Neon, and Claude

2026-09-17

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

Security researcher Gal Weizman of Forever Security disclosed BragJack, a proof-of-concept attack technique showing that a single malicious browser extension, requesting only two permissions that are common among ad blockers and similar utilities, could hijack the built-in AI assistants of five separate products: Google Chrome's Gemini Live panel, Perplexity Comet, Microsoft Edge, Opera Neon, and the Claude for Chrome extension [1][2]. Rather than relying on prompt injection or guardrail bypass, BragJack exploits a shared architectural weakness in how these browsers route trust between an extension, the browser's privileged AI surface, and the vendor's web application, allowing a low-privilege extension to impersonate the domain the AI assistant is designed to obey [1][3]. Two of the five findings, affecting Chrome and Microsoft Edge, were formalized as CVE-2026-0628 and CVE-2026-55945 and have been patched; the remaining three, in Comet, Opera Neon, and Claude for Chrome, were remediated by their vendors without a public CVE assignment [1][4]. All five vendors paid bug bounties totaling roughly $20,500, and as of this writing there is no public evidence that BragJack techniques have been used in real-world attacks; the research followed a coordinated-disclosure process — Weizman reported each finding to the affected vendor before publication — and the underlying flaws have since been fixed [1][5]. The disclosure nonetheless illustrates a structural risk in agentic browser design: because every major vendor built its AI assistant on the same extension permission model, a single technique generalized across otherwise-competing products, a pattern CSA has now observed for a second time since Chrome's Gemini panel was first shown to be exploitable in this way in March 2026 [3] [9].

Background

Over the past year, Chrome, Edge, Opera, Perplexity, and Anthropic have each shipped an AI assistant that lives inside the browser rather than in a separate tab or application. These assistants are typically described as having a "body" and a "brain": a privileged front-end component running inside the browser that can see the screen, read the page, and take actions such as clicking, typing, or opening files, paired with a "brain" consisting of a large language model that runs on the vendor's servers and issues instructions back to that front-end component [1]. This division of labor is what allows the assistant to

act on a user's behalf, but it also means the front-end component must trust something about the messages it receives so that it can distinguish legitimate instructions from the vendor's server from anything else running in the browser.

Weizman's research began with a narrower finding. In March 2026, working at the time as a researcher affiliated with Palo Alto Networks' Unit 42, he disclosed GlicJack, a vulnerability in Chrome's Gemini Live panel, accessible at the internal address `chrome://glic`, that had not been included in the browser's extension content-script blocklist [3][6]. Because the panel was omitted from that list, an extension holding only the declarativeNetRequest permission, an API designed for legitimate uses such as ad blocking and privacy filtering, could inject JavaScript directly into the panel and impersonate Google's own Gemini web application at `gemini.google.com` [3][6]. Google assigned this flaw CVE-2026-0628 with a CVSS score of 8.8, likely reflecting the panel's access to the camera, microphone, local files, and screenshot capabilities, and shipped a fix in Chrome 143.0.7499.192 in early January 2026 after Weizman reported the issue to Google in October 2025 [3][6].

By September 2026, Weizman, now researching independently under the banner Forever Security, had generalized the same underlying idea, that a browser extension can impersonate the domain an AI assistant is built to trust, into a technique he named BragJack, and applied it against four additional products beyond Chrome [1][2]. The consistency of the finding across five independently engineered browsers, built by four different companies, suggests the vulnerability class is a consequence of a shared design pattern in agentic browser architecture rather than an implementation mistake specific to any single vendor. This is consistent with CSA's own earlier assessment of the original Chrome panel flaw, which concluded that the structural challenge of embedding AI capability directly in the browser while preserving the isolation guarantees that browser security models are built around would not be resolved by a single patch cycle [9].

Security Analysis

BragJack's core insight is that an extension needs only two widely granted permissions to attempt this class of attack: one that allows modifying the content of web pages, a capability common to ad blockers, and one that allows using the declarativeNetRequest API to alter network requests and responses, a capability common to privacy and ad-filtering extensions as well [1][2]. Neither permission by itself looks alarming to a user reviewing an extension's install prompt, and neither is unique to any single browser; this arguably is what makes the technique broadly applicable rather than a narrow exploit against one product's implementation choices. With those two permissions, an attacker can weaken or bypass the

Content Security Policy protecting the page the AI assistant's front-end loads, redirect the JavaScript resources that page depends on to attacker-controlled servers, and inject code that issues commands directly to the AI agent as though those commands had come from the trusted vendor domain [4][5].

The five affected products did not share an identical implementation, and the severity of what an attacker could accomplish varied accordingly. In Chrome, the same `chrome://glic` panel weakness underlying CVE-2026-0628 permitted reading local files, activating the camera and microphone, capturing screenshots, and leaking data from the user's browser profile [1][2]. Comet, Perplexity's agentic browser, was vulnerable through a similar combination of permissions and involved a testing subdomain, `testing.perplexity.com`, that broadened the technique into full agent command control alongside file reading, browsing-history access, and profile leakage [1][2]. Microsoft Edge required a more specific setup, exploiting a timing race condition through a compromised marketing page to achieve control of the AI agent; this variant was assigned CVE-2026-55945 with a comparatively low CVSS score of 4.2, likely reflecting that exploitation required local or adjacent access and some pre-existing privilege rather than being remotely triggerable at scale [1][7]. Opera Neon was compromised through direct code injection on the `opera.com` domain, again yielding agent command capability [1][2]. Claude for Chrome, notably, was the only product in the study that is itself a browser extension rather than a full browser; Weizman characterized this variant as the least serious of the five because it involved one extension abusing another rather than an extension abusing the browser's own privileged surface, though it still allowed an attacker to command the AI agent [1][2].

The Claude for Chrome finding disclosed under BragJack is a distinct vulnerability from the synthetic-click flaw in the same product that Manifold Security researchers disclosed separately in May 2026 and that CSA analyzed in its own research note that July [8]. That earlier flaw allowed a co-installed extension to forge a click event and trigger a pre-approved agent task by exploiting Claude for Chrome's failure to check the browser's `Event.isTrusted` property [8]. BragJack's Claude for Chrome variant instead exploits inter-extension communication to issue commands to the AI agent directly. Together, the two disclosures indicate that Claude for Chrome's extension-adjacent trust boundary has now been probed successfully by at least two independent research efforts within a five-month span, using different technical mechanisms but arriving at a similar outcome: an unrelated, co-installed extension gaining influence over the AI agent's behavior.

Across all five cases, the underlying lesson is the same one CSA articulated in its own analysis of the original Chrome disclosure: putting an AI agent inside the browser reopens a path that extension isolation models are generally designed to close, in which a low-privilege component, the extension, can reach a high-privilege surface, the AI agent's command channel, that was not designed with that adjacency in mind [1][9]. Traditional browser extension security models assume that extensions are mutually distrustful and that the browser's own chrome, the trusted user-interface surface that is not a web page, is categorically off-limits to extension code. Embedding an AI agent with camera, microphone,

file-system, and account access inside that same browser chrome, while still allowing users to install arbitrary third-party extensions with page-modification and network-interception permissions, creates exactly the adjacency BragJack exploits. No CVE was assigned for three of the five findings; this is consistent with, though does not by itself prove, the flaw being architectural rather than a discrete implementation bug — the vendors did not need to find a coding error so much as recognize that the AI assistant's front-end had never been explicitly protected from co-resident extension code.

Recommendations

Immediate Actions

Organizations should confirm that Chrome installations are updated to at least version 143.0.7499.192, the release in which CVE-2026-0628 was fixed [3], and that Microsoft Edge installations are updated to at least version 150.0.4078.48, the release in which CVE-2026-55945 was fixed [7]; these are the only two BragJack variants with disclosed CVE identifiers and confirmed patch versions. Security teams should audit which browser extensions are installed across managed endpoints, with particular attention to any extension requesting both page-content modification and declarativeNetRequest permissions, and should remove or restrict extensions that request this combination without a clear justification tied to their advertised function. Enterprises that have deployed Perplexity Comet, Opera Neon, or Claude for Chrome should confirm with those vendors that the specific fixes described in the September 2026 disclosure have been applied, since these three variants were remediated without a public CVE and therefore lack an independently verifiable patch-version marker.

Short-Term Mitigations

Enterprises should extend existing extension allowlisting and permission-review programs, which are typically built to govern data-loss and privacy risk, to explicitly cover the AI assistants built into browsers, treating the AI agent's command channel as a sensitive asset that a review process must protect rather than an invisible internal feature of the browser. Where feasible, organizations should limit the use of agentic browser AI features to dedicated, low-privilege browser profiles that have no other extensions installed, reducing the practical attack surface even if a BragJack-style flaw exists in an unpatched form. Security monitoring teams should treat unusual AI-agent-initiated actions, such as unexpected file access, camera or microphone activation, or navigation to unfamiliar destinations, as indicators worth investigating, since these are the behaviors a successful BragJack-style hijack would ultimately produce regardless of which of the five browsers is involved.

Strategic Considerations

Because BragJack demonstrates that this vulnerability class recurred across five independently built products in roughly six months, organizations evaluating agentic browser adoption should treat this as an ongoing category risk rather than a set of individually resolved incidents, and should ask prospective and existing vendors how they have hardened the specific trust boundary between third-party extensions and their AI agent's command surface. Enterprise security teams should also weigh the operational reality that enterprise browsing routinely involves third-party extensions, meaning agentic browser features can inherit the security posture of whichever extension a user has installed, regardless of how trustworthy it is, a dependency that is difficult to fully mitigate through vendor-side patching alone. Finally, this disclosure supports a broader industry conversation about whether AI agents embedded in general-purpose browsers should be granted camera, microphone, and file-system access by default, or whether such capabilities should require a more deliberate, narrowly scoped trust model given how readily a common extension permission set can be turned against them.

CSA Resource Alignment

BragJack extends a specific vulnerability class that CSA has already analyzed in detail. CSA's research note, Browser-Integrated AI Panel Hijack: CVE-2026-0628 and the Emerging Attack Surface of Embedded AI, examined the original Chrome Gemini Live flaw that Weizman disclosed in March 2026 under the name GlicJack, and concluded that the underlying design tension between embedding AI capability directly in the browser and preserving extension isolation would not be resolved by patching a single instance of the flaw [9]. BragJack's expansion of the same technique to Comet, Edge, Opera Neon, and Claude for Chrome five months later directly confirms that assessment. CSA's research note on the Claude for Chrome Synthetic Click Flaw is also directly relevant, since it analyzed a separate but related trust-boundary failure in the same product, reinforcing that Claude for Chrome's exposure to co-installed extensions has now been demonstrated through at least two independent mechanisms [8].

Because the root cause in all five cases is a failure to establish trust between a low-privilege extension and a high-privilege AI agent surface, this incident also falls squarely within the scope of CSA's Zero Trust Principles and Guidance for Identity and Access Management (IAM), which argues that access decisions should be based on continuous verification of the requesting component rather than on implicit trust granted by co-location within the same application or platform; browsers that grant an AI agent implicit trust in messages appearing to originate from its own vendor domain, without verifying the provenance of the code injecting those messages, run counter to this principle in a way BragJack makes concrete. Organizations mapping this incident to a broader control framework should also reference the AI Controls Matrix (AICM) v1.1, whose agentic AI and identity and access management domains address

authentication of components communicating with AI agents and isolation of privileged AI capabilities from lower-trust system components, a pattern applicable to the browser-extension adjacency BragJack exploits. CSA's overview of the framework, Agentic AI Threat Modeling Framework: MAESTRO, further provides a layered model for reasoning about this class of cross-boundary trust failure between an agentic AI system's orchestration layer and the environment hosting it.

References

[1] The Hacker News. "One Extension Could Hijack AI Assistants Across Chrome, Comet, Edge, Opera Neon and Claude." The Hacker News, September 16, 2026.

[2] Gblock. "BragJack: One Extension Hijacks 5 Browser AI Assistants." Gblock, September 2026.

[3] The Hacker News. "New Chrome Vulnerability Let Malicious Extensions Escalate Privileges via Gemini Panel." The Hacker News, March 2, 2026.

[4] OffSeq Threat Radar. "BragJack - $20K in bounty rewards from Anthropic, Perplexity, Google, Microsoft and Opera." OffSeq, September 2026.

[5] Strix. "CVE-2026-55945: Edge Chromium Race Condition (CVSS 4.2)." Strix, 2026.

[6] Palo Alto Networks Unit 42. "Taming Agentic Browsers: Vulnerability in Chrome Allowed Extensions to Hijack New Gemini Panel." Palo Alto Networks, 2026.

[7] OpenCVE. "CVE-2026-55945." OpenCVE, 2026.

[8] Cloud Security Alliance AI Safety Initiative. "Claude for Chrome: Synthetic Click Flaw Lets Extensions Hijack AI Actions." Cloud Security Alliance, July 18, 2026.

[9] Cloud Security Alliance AI Safety Initiative. "Browser-Integrated AI Panel Hijack: CVE-2026-0628 and the Emerging Attack Surface of Embedded AI." Cloud Security Alliance, March 9, 2026.

[10] Cloud Security Alliance. "Zero Trust Principles and Guidance for Identity and Access Management (IAM)." Cloud Security Alliance, 2024.

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

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

