INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-03-29 18:35 UTC

ShadowPrompt: How a Misconfigured Trust Boundary Turned Claude’s Browser Extension into a Zero-Click Attack Vector

CVE VULNERABILITY | **HIGH** | CVSS 7.5

SCC Item ID	SCC-CVE-2026-0024
Type	CVE Vulnerability
Severity	HIGH
CVSS Base Score	7.5
Affected Products	Anthropic Claude Chrome Extension (patched in v1.0.41); Arkose Labs CAPTCHA component hosted at a-cdn.claude.ai (XSS resolved 2026-02-19)
Published	2026-03-26
Discovery Source	Rss

Executive Summary

A combined vulnerability in the Anthropic Claude Chrome Extension allowed any malicious website to silently inject instructions into a user's Claude AI assistant without any click or interaction. The flaw combined an overly permissive trust boundary in the extension with a cross-site scripting weakness in a third-party CAPTCHA component embedded on the claude.ai subdomain. If exploited before patching, an attacker could have stolen credentials, accessed conversation history, or triggered autonomous actions such as sending emails on behalf of the victim, with no visible indication to the user.

Technical Analysis

The vulnerability combines two weaknesses in the Anthropic Claude Chrome Extension (patched in v1.0.41 following responsible disclosure on 2025-12-27). First, the extension's subdomain allowlist (CWE-346: Origin Validation Error) granted implicit trust to all subdomains under claude.ai, including a-cdn.claude.ai, an endpoint controlled by the third-party Arkose Labs CAPTCHA service. Second, a DOM-based XSS flaw (CWE-79) in the Arkose Labs component at a-cdn.claude.ai allowed attacker-controlled pages to inject arbitrary JavaScript into that trusted context. Combined, these flaws enabled indirect prompt injection (MITRE T1059.007, T1534): a malicious page could deliver arbitrary prompts to the Claude assistant without user interaction. Downstream risks include session token theft (T1539), browser session hijacking (T1185), and unauthorized autonomous actions on behalf of the victim (T1557). CWE-20 (Input Validation) and CWE-693 (Protection Mechanism Failure) are also mapped. The Arkose Labs XSS was independently resolved on 2026-02-19. No CVE identifier

has been confirmed for this vulnerability. CVSS base score is assessed at 7.5 (High). EPSS data is not available. The item is not listed in the CISA KEV catalog.

Action Checklist

1. Step 1: Containment. Immediately verify that all enterprise endpoints running the Claude Chrome Extension are on v1.0.41 or later. If any instances remain on earlier versions, disable the extension via Chrome policy (ExtensionInstallBlocklist or managed browser controls) until patching is confirmed. Verify via Chrome Management console or MDM query to confirm v1.0.41 deployment; do not assume silent update completed.
2. Step 2: Detection. Review Chrome extension audit logs and browser management console reports for extension version inventory. Query endpoint management tooling (e.g., Chrome Browser Cloud Management, Jamf, Intune) for installed extension versions. Look for anomalous outbound requests to a-cdn.claude.ai in proxy and DNS logs, particularly from endpoints where the extension was active prior to the patch. No confirmed IOC signatures are available at this time.
3. Step 3: Eradication. Force-update the Claude Chrome Extension to v1.0.41 or later across all managed endpoints via Chrome policy or MDM push. Confirm the Arkose Labs XSS remediation (resolved 2026-02-19) is reflected in the current extension version before re-enabling in any restricted environments. Remove earlier versions explicitly; do not assume silent update completed.
4. Step 4: Recovery. After updating, validate extension version via Chrome Management console or endpoint query. Monitor browser telemetry for any residual connections to a-cdn.claude.ai from pre-patch activity windows. Review Claude conversation history for any sessions that may have occurred on affected versions during the exposure window (2025-12-27 disclosure to v1.0.41 deployment) for signs of injected or anomalous prompts.
5. Step 5: Post-Incident. Audit all browser extensions in the enterprise for subdomain trust configurations and third-party component dependencies. Establish a policy requiring security review of AI assistant browser extensions before deployment, with version-pinning and forced-update controls. Map this gap to CWE-346 and CWE-693 control reviews in your GRC program. Incorporate indirect prompt injection as an attack scenario in threat models for any AI-integrated tooling.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to CISO and legal/privacy counsel if proxy or DNS logs confirm any endpoint loaded content from a-cdn.claude.ai during the 2025-12-27 through v1.0.41 deployment exposure window AND Claude conversation history shows anomalous outputs, credential-related prompts, or autonomous actions not initiated by the user — this combination indicates confirmed exploitation and may trigger breach notification obligations under GDPR, CCPA, or HIPAA depending on data types accessible via the compromised Claude sessions.

Recovery Notes	Re-enable the Claude Chrome Extension only after confirming v1.0.41 or later via endpoint query — not auto-update status alone — and after validating that the updated manifest.json no longer grants the 'a-cdn.claude.ai' subdomain the same trust level as 'claude.ai' proper. Monitor proxy logs for outbound connections to 'a-cdn.claude.ai' from all endpoints for a minimum of 30 days post-recovery, as any such connections from a patched extension would be anomalous and warrant immediate re-investigation. If any user accounts show evidence of conversation exfiltration or autonomous Claude actions during the exposure window, treat those accounts as compromised, rotate associated API keys and OAuth tokens, and review downstream systems those accounts had access to via Claude's tool-use or browser automation capabilities.
Forensic Artifacts	Chrome extension manifest.json at '%LOCALAPPDATA%\Google\Chrome\User Data\Default\Extensions\npdnfjpoanmonaoeodkbbpetfimnaeif\[version]\manifest.json' — preserves the pre-patch 'host_permissions' scope showing the CWE-346 trust boundary failure that allowed a-cdn.claude.ai XSS content to execute with claude.ai extension privileges | Proxy/web gateway access logs filtered to 'a-cdn.claude.ai' during the 2025-12-27 to patch-deployment exposure window — the Arkose Labs CAPTCHA component hosted here was the XSS delivery point; POST requests or abnormal query parameters in these logs are the primary indicator of active exploitation | Claude.ai conversation history exports for accounts used on affected endpoints during the exposure window — indirect prompt injection payloads would appear as unexpected instruction-following behavior, system-directive outputs, or autonomous actions (credential requests, data exfiltration commands) interleaved in user conversation turns | Chrome 'History' SQLite database at '%LOCALAPPDATA%\Google\Chrome\User Data\Default\History' — SQL query 'SELECT url, last_visit_time FROM urls WHERE url LIKE "%a-cdn.claude.ai%"' establishes which endpoints loaded the vulnerable CAPTCHA component and correlates session timing to the exposure window | Claude.ai account activity logs for OAuth token issuance, API key creation, and conversation sharing events during the exposure window — autonomous execution of these actions without user initiation maps to MITRE ATT&CK T1059.007 (JavaScript execution via XSS-delivered payload) and T1539 (Steal Web Session Cookie) as plausible post-injection objectives

Per-Action IR Details

Step 1: Containment — Immediately verify that all enterprise endpoints running the Claude Chrome Extension are on v1.0.41 or later. If any instances remain on earlier versions, disable the extension via Chrome policy (ExtensionInstallBlocklist or managed browser controls) until patching is confirmed. Do not rely on auto-update confirmation alone.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: short-term containment to stop further damage before eradication is complete

Controls: NIST IR-4 (Incident Handling), NIST CM-7 (Least Functionality) — restrict browser extension execution to approved, versioned software only, CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 4.6 (Securely Manage Enterprise Assets and Software)

Compensating: For teams without centralized Chrome management: run 'Get-ItemProperty HKLM:\SOFTWARE\Google\Chrome\Extensions* | Select Name, Version' on Windows endpoints (PowerShell) or 'find /home -name 'manifest.json' -path '*/Extensions/*' | xargs grep -l 'claude' | xargs grep 'version'' on Linux to inventory installed extension versions. Manually set 'ExtensionInstallBlocklist' in a local GPO with the Claude extension ID (npdnfjpoanmonaoeodkbbpetfimnaeif) blocked until v1.0.41 is confirmed. A 2-person team can script this check across a flat network using psexec or Ansible in under an hour.

Evidence: Before disabling the extension, capture: (1) Chrome extension manifest at '%LOCALAPPDATA%\Google\Chrome\User Data\Default\Extensions\npdnfjpoanmonaoeodkbbpetfimnaeif\[version]\manifest.json' to preserve the pre-patch version number and declared permissions (particularly the 'claude.ai' host_permissions and content_script match patterns that enabled the overly broad trust boundary); (2) Chrome 'Local State' and 'Preferences' JSON files at '%LOCALAPPDATA%\Google\Chrome\User Data\' to record extension install timestamps and last-update checks; (3) Chrome browser process memory dump if any active Claude sessions exist — indirect prompt injection payloads delivered via the XSS vector would reside in renderer process memory tied to a-cdn.claude.ai frames.

Step 2: Detection — Review Chrome extension audit logs and browser management console reports for extension version inventory. Query endpoint management tooling (e.g., Chrome Browser Cloud Management, Jamf, Intune) for installed extension versions. Look for anomalous outbound requests to a-cdn.claude.ai in proxy and DNS logs, particularly from endpoints where the extension was active prior to the patch. No confirmed IOC signatures are available at this time.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: correlate indicators across log sources to determine scope and establish exposure window (2025-12-27 through v1.0.41 deployment date)

Controls: NIST IR-5 (Incident Monitoring), NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Without a SIEM: (1) Extract proxy logs (Squid/Bluecoat/Zscaler) and run 'grep -i "a-cdn.claude.ai" access.log | awk '{print $1,$7,$8}' | sort | uniq -c | sort -rn' to surface high-frequency requests that may indicate injection loop activity; (2) Use osquery with 'SELECT name, version, identifier FROM chrome_extensions WHERE identifier = 'npdnfjpoanmonaoeodkbbpetfimnaeif';' to pull extension version from live endpoints; (3) Deploy the Sigma rule targeting Chrome extension web request events (sigma/rules/application/browser/browser_extension_susp_communication.yml as a starting template) adapted to filter on host 'a-cdn.claude.ai' and referrer patterns from claude.ai subdomain iframes; (4) Query DNS resolver logs for 'a-cdn.claude.ai' resolution volume per endpoint over the exposure window.

Evidence: Capture before analysis concludes: (1) Proxy/web gateway logs showing HTTP requests to 'a-cdn.claude.ai' — the Arkose Labs CAPTCHA component XSS vector — filtered to the exposure window (2025-12-27 to patch deployment); flag any POST requests or unexpected query parameters that could carry injected prompt payloads exfiltrated via the XSS; (2) DNS query logs for 'a-cdn.claude.ai' and any subdomains, per endpoint, to establish which machines loaded the malicious CAPTCHA component; (3) Chrome browser history ('History' SQLite DB at '%LOCALAPPDATA%\Google\Chrome\User Data\Default\History') from affected endpoints — query 'SELECT url, title, last_visit_time FROM urls WHERE url LIKE "%claude.ai%" OR url LIKE "%a-cdn.claude.ai%"' to reconstruct which sessions were active during the exposure window; (4) Claude conversation history exports via claude.ai account settings for accounts used on affected endpoints during the exposure window — look for conversation turns containing unexpected instructions, system-level directives, or outputs inconsistent with user intent that would indicate a silently injected prompt.

Step 3: Eradication — Force-update the Claude Chrome Extension to v1.0.41 or later across all managed endpoints via Chrome policy or MDM push. Confirm the Arkose Labs XSS remediation (resolved 2026-02-19) is reflected in the current extension version before re-enabling in any restricted environments. Remove earlier versions explicitly; do not assume silent update completed.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: eliminate the vulnerability from all affected systems and verify removal before recovery; silent auto-update cannot substitute for confirmed remediation

Controls: NIST SI-2 (Flaw Remediation), NIST CM-3 (Configuration Change Control), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 2.3 (Address Unauthorized Software)

Compensating: For teams without MDM: (1) Push Chrome ExtensionInstallForcelist GPO entry pinning the Claude extension to the v1.0.41 update URL — set 'ExtensionInstallForcelist' = 'npdnfjpoanmonaoeodkbbpetfimnaeif;https://clients2.google.com/service/update2/crx' to force Chrome to pull the current store version on next browser launch; (2) Verify remediation by re-running the osquery extension version check post-restart; (3) Manually inspect the updated manifest.json 'content_scripts' and 'host_permissions' sections to confirm the overly broad trust boundary granting access to all claude.ai subdomains (including a-cdn.claude.ai) has been scoped down in v1.0.41 — if permissions remain identical to pre-patch versions, treat the update as incomplete and escalate to Anthropic.

Evidence: Before forcing update, preserve the vulnerable extension package: copy the full extension directory at '%LOCALAPPDATA%\Google\Chrome\User Data\Default\Extensions\npdnfjpoanmonaoeodkbbpetfimnaeif\[pre-patch version]\' to offline forensic storage — this preserves the manifest.json with the flawed host_permissions scope and the content_script injection logic that trusted all claude.ai subdomains (the CWE-346 origin validation failure), which is required evidence if a breach notification or post-incident review is needed. Hash the directory contents (SHA-256) before and after removal.

Step 4: Recovery — After updating, validate extension version via Chrome Management console or endpoint query. Monitor browser telemetry for any residual connections to a-cdn.claude.ai from pre-patch activity windows. Review Claude conversation history for any sessions that may have occurred on affected versions during the exposure window (2025-12-27 disclosure to v1.0.41 deployment) for signs of injected or anomalous prompts.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: restore systems to normal operation with confidence that the vulnerability is resolved; monitor for recurrence during the observation period covering the full exposure window

Controls: NIST IR-4 (Incident Handling), NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST SI-7 (Software, Firmware, And Information Integrity), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Without browser telemetry tooling: (1) Use Wireshark or tcpdump on a representative sample of endpoints post-update, filtering on 'host a-cdn.claude.ai', to confirm no residual connections originate from the updated extension (the patched version should no longer grant the CAPTCHA subdomain the trust elevation that enabled the XSS chaining); (2) Script a daily cron job or scheduled task for 30 days post-recovery running 'osquery' extension version checks to catch any regression from unexpected extension rollback or profile sync events; (3) For conversation history review, instruct users to export their Claude.ai conversation history (Settings → Data Export) and manually inspect turns from the 2025-12-27 through patch-deployment window for injected system-level instructions or outputs referencing credential access, conversation exfiltration, or autonomous task execution — hallmarks of an indirect prompt injection payload.

Evidence: During recovery monitoring, collect: (1) Post-update proxy logs for 'a-cdn.claude.ai' traffic — any continued requests from endpoints confirmed on v1.0.41 would indicate either update failure or a new injection vector and require immediate re-escalation; (2) Claude.ai account activity logs (accessible via account settings) for API key generation events, OAuth token issuance, or conversation sharing actions that occurred during the exposure window without explicit user initiation — these would indicate the injection payload successfully triggered autonomous actions (MITRE ATT&CK T1059.007 — Command and Scripting Interpreter: JavaScript, via the XSS-delivered payload); (3) Browser sync logs if Google account sync is enabled — verify that a compromised profile state was not synced to additional devices during the exposure window, which would extend the blast radius beyond initially identified endpoints.

Step 5: Post-Incident — Audit all browser extensions in the enterprise for subdomain trust configurations and third-party component dependencies. Establish a policy requiring security review of AI assistant browser extensions before deployment, with version-pinning and forced-update controls. Map this gap to CWE-346 and CWE-693 control reviews in your GRC program. Incorporate indirect prompt injection as an attack scenario in threat models for any AI-integrated tooling.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: lessons learned, policy updates, and detection improvements to prevent recurrence; update threat models to include AI-specific attack surfaces

Controls: NIST IR-8 (Incident Response Plan) — update to include AI assistant browser extension scope and indirect prompt injection scenarios, NIST RA-3 (Risk Assessment) — reassess risk for all deployed browser extensions with AI/LLM backend access, NIST SI-10 (Information Input Validation) — apply to AI assistant integrations as a control for prompt injection boundaries, NIST CM-7 (Least Functionality) — enforce extension allowlisting with version-pinning, CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 2.1 (Establish and Maintain a Software Inventory) — extend to include browser extension inventory with version tracking, CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory) — include browser extension profiles per endpoint

Compensating: For teams without a GRC platform: (1) Use a spreadsheet-based extension audit template — enumerate all installed Chrome extensions via osquery ('SELECT name, identifier, version, permissions FROM chrome_extensions;') exported to CSV, then manually flag any extension whose manifest.json 'host_permissions' includes wildcard subdomain patterns (e.g., '*://*.claude.ai/*' or '*://*.service.ai/*') and third-party CDN dependencies not controlled by the extension vendor; (2) Write a YARA rule targeting Chrome extension manifest.json files to flag overly broad subdomain trust patterns for future extension reviews; (3) Document CWE-346 (Origin Validation Error) and CWE-693 (Protection Mechanism Failure) as recurring risk themes in your risk register, linking them to any other deployed browser extensions that embed third-party scripts from subdomains of trusted origins; (4) Add 'indirect prompt injection via browser extension XSS chaining' as a named scenario in your next tabletop exercise using this incident as the scenario seed.

Evidence: Post-incident artifacts to retain for GRC and potential regulatory review: (1) Full incident timeline documenting the exposure window (2025-12-27 disclosure through v1.0.41 deployment date) with per-endpoint patch confirmation timestamps — required for breach notification threshold analysis if credential or conversation data was accessed; (2) The preserved pre-patch manifest.json from Step 3, annotated to show the specific 'host_permissions' and 'content_scripts' entries constituting the CWE-346 trust boundary failure — serves as evidence artifact for GRC control gap documentation; (3) Proxy and DNS log exports from the exposure window, retained per AU-11 (Audit Record Retention) requirements, to support any future legal hold or regulatory inquiry into whether the XSS on a-cdn.claude.ai was actively exploited against enterprise users.

Detection Guidance

No confirmed IOCs (IPs, hashes, or signatures) are available for this vulnerability at the time of this report. Detection relies on version and behavioral indicators. Query your endpoint or browser management platform for any Claude Chrome Extension instances running versions below v1.0.41. In proxy and DNS logs, flag connections to a-cdn.claude.ai originating from browser processes during the exposure window (approximately 2025-12-27 through your confirmed patch deployment date) for additional review. In Claude conversation logs or session data, look for prompts or responses that appear inconsistent with the user's direct input, particularly instructions involving credential submission, email composition, or external service authentication. Behavioral anomalies in browser-integrated AI sessions (unexpected actions, unfamiliar prompt patterns) should be treated as potential prompt injection indicators pending further investigation. Escalation to IR is warranted if session compromise is suspected on a production system.

Indicators of Compromise

Type	Value	Context	Confidence
DOMAIN	`a-cdn.claude.ai`	Third-party Arkose Labs CAPTCHA endpoint trusted by Claude Chrome Extension subdomain allowlist; XSS delivery point in exploit chain. Legitimate domain — flag unexpected or high-volume connections from browser processes during the pre-patch exposure window only.	MEDIUM

Framework Mappings

MITRE-ATTACK

- **T1557** — Adversary-in-the-Middle
- **T1059** — Command and Scripting Interpreter
- **T1539** — Steal Web Session Cookie
- **T1185** — Browser Session Hijacking
- **T1190** — Exploit Public-Facing Application
- **T1059.007** — JavaScript
- **T1534** — Internal Spearphishing

NIST-800-53R5

- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-7** — Software, Firmware, and Information Integrity
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SI-10** — Information Input Validation
- **SR-2** — Supply Chain Risk Management Plan

OWASP-TOP10-2021

- **A03:2021** — Injection

CIS-V8

- **16.10**
- **6.3** — Require MFA for Externally-Exposed Applications
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management
- **15.1** — Establish and Maintain an Inventory of Service Providers

ISO-27001-2022

- **A.8.28** — Secure coding
- **A.8.26** — Application security requirements
- **A.8.8** — Management of technical vulnerabilities
- **A.5.21** — Managing information security in the ICT supply chain
- **A.5.23** — Information security for use of cloud services

HIPAA-SECURITY

- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures
- **CC9.2** — Manages risks associated with vendors and business partners
- **CC6.3** — Authorizes, modifies, or removes access

NIST-CSF-2

- **GV.SC-01** — Cybersecurity supply chain risk management program

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1557	Adversary-in-the-Middle	Credential-Access
T1059	Command and Scripting Interpreter	Execution
T1539	Steal Web Session Cookie	Credential-Access
T1185	Browser Session Hijacking	Collection
T1190	Exploit Public-Facing Application	Initial-Access
T1059.007	JavaScript	Execution
T1534	Internal Spearphishing	Lateral-Movement

Sources

Source	URL	Tier
Security News	https://thehackernews.com/2026/03/claude-extension-flaw-enabled-zer...	T3
Reverse engineering Claude's CVE-2026-2796 exploit	https://red.anthropic.com/2026/exploit/	T1

Source	URL	Tier
New Security Risks Emerge from Anthropic's Claude Chrome ...	https://www.itcpeacademy.org/blog/news-010526-new-security-risks-em...	T3
New Zero-Click Flaw in Claude Extensions, Anthropic Declines Fix	https://www.infosecurity-magazine.com/news/zeroclick-flaw-claude-dxt/	T3
Anthropic just dropped Claude for Chrome – AI that fully controls ...	https://www.reddit.com/r/ClaudeAI/comments/1prcypb/anthropic_just_d...	T3

Generated 2026-03-29 18:35 UTC by TJS Security Command Center