INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-06 08:41 UTC

GHSA-pmwg-cvhr-8vh7: Axios: Incomplete Fix for CVE-2025-62718, NO_PROXY Protection Bypassed via RFC

CVE VULNERABILITY | HIGH | CVSS 7.4

SCC Item ID	SCC-CVE-2026-0128
Type	CVE Vulnerability
CVE ID	CVE-2026-42043
Severity	HIGH
CVSS Base Score	7.4
EPSS Score	0.0004 (13th percentile)
Affected Products	axios (npm) 1.15.0
Published	2026-05-05T00:20:58Z
Discovery Source	Osv

Executive Summary

A security flaw in Axios 1.15.0, a widely used JavaScript HTTP library, allows traffic that should stay internal to be routed through external proxy servers by exploiting an incomplete network address check. Organizations running applications built on this library version may unknowingly expose internal services or sensitive API traffic to unintended proxy endpoints. The business risk is unauthorized data exposure through misdirected HTTP requests, particularly in environments where proxy controls are used to enforce network segmentation.

Technical Analysis

CVE-2026-42043 (GHSA-pmwg-cvhr-8vh7) is an incomplete fix for CVE-2025-62718 in axios npm package version 1.15.0. The prior fix blocked NO_PROXY bypass only for the canonical loopback address 127.0.0.1, failing to account for the full RFC 1122 loopback subnet 127.0.0.0/8. An attacker or misconfigured application can address requests to alternate loopback addresses (e.g., 127.0.0.2 through 127.255.255.254) to bypass NO_PROXY rules and route those requests through a configured proxy. CVSS base score: 7.4 (High). CWE-184 (Incomplete List of Disallowed Inputs) and CWE-20 (Improper Input Validation) apply. MITRE ATT&CK mappings: T1090 (Proxy) and T1071.001 (Application Layer Protocol: Web Protocols). EPSS score is 0.044% (13th percentile), indicating low current exploitation activity. Not listed in CISA KEV. Full CVSS vector

details were not available in source metadata; base score sourced from OSV advisory. Linux distributions are flagged as unpatched in current Tenable plugin data. No confirmed patch version was extractable from available source metadata; verify the latest axios release on the official npm registry or GitHub repository.

Action Checklist

1. Step 1: Containment, Identify all applications and services in your environment that depend on axios 1.15.0 by running 'npm list axios' or scanning your software bill of materials (SBOM). Temporarily restrict outbound proxy routing for any service confirmed to use this version until patched.
2. Step 2: Detection, Query your dependency inventory, CI/CD pipeline artifacts, and container image registries for axios@1.15.0. In proxy access logs, look for unexpected requests originating from loopback addresses in the 127.0.0.0/8 range that passed through your proxy rather than being blocked. Review Node.js application logs for HTTP requests targeting 127.x.x.x addresses other than 127.0.0.1.
3. Step 3: Eradication, Upgrade axios to the latest version available on npm that explicitly addresses CVE-2026-42043 and GHSA-pmwg-cvhr-8vh7. Confirm the release notes or changelog reference the NO_PROXY loopback subnet fix before deploying. Run 'npm audit' post-upgrade to verify no residual advisories remain for axios.
4. Step 4: Recovery, After upgrading, validate that NO_PROXY rules correctly block requests to the full 127.0.0.0/8 range by running integration tests that target alternate loopback addresses. Monitor proxy logs for 48 hours post-remediation for any anomalous loopback-sourced traffic that bypasses NO_PROXY controls.
5. Step 5: Post-Incident, Review your patch management process for third-party npm dependencies, specifically the gap between incomplete-fix CVEs and full remediation. Evaluate whether your SBOM tooling flags incomplete fixes (CWE-184) as distinct from new vulnerabilities. Consider adding automated checks for the full loopback subnet in proxy configuration validation.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to security leadership and initiate breach notification review if proxy access logs confirm any 127.0.0.0/8-sourced requests successfully routed through an external or untrusted proxy, as this indicates actual data exfiltration exposure that may trigger regulatory notification obligations under GDPR, HIPAA, or applicable state breach notification laws.
Recovery Notes	After deploying the patched axios version, verify remediation by running targeted integration tests against 127.0.0.2 and 127.255.255.254 loopback addresses with NO_PROXY set to 127.0.0.0/8 — successful proxy bypass of these addresses indicates the fix is incomplete. Monitor Squid or nginx proxy access logs for 48 hours using a 15-minute cron grep for CONNECT or GET requests sourced from any 127.x.x.x address to confirm no residual bypass traffic. Rebuild and redeploy all container images that bundled axios 1.15.0 from scratch rather than patching in-place, as in-place node_modules upgrades in containers can leave stale cached module state.

Forensic Artifacts	Proxy access logs (Squid: /var/log/squid/access.log; nginx: /var/log/nginx/access.log) filtered for requests with source or target in the 127.0.0.0/8 range excluding 127.0.0.1 — these entries are the direct forensic indicator of successful NO_PROXY bypass exploitation via CVE-2026-42043 | Node.js application stdout/stderr logs (journald or /var/log//) containing outbound HTTP request records targeting 127.x.x.x addresses, which would indicate the application invoked the vulnerable axios proxy-resolution code path with an alternate loopback address | Pre-patch snapshots of node_modules/axios/lib/helpers/ source files (particularly proxy-handling utilities) with SHA-256 hashes documenting the presence of the incomplete subnet-check code in axios 1.15.0 at time of incident | package-lock.json and node_modules/.package-lock.json from all affected application deployments, preserving the exact resolved axios 1.15.0 dependency version and its transitive dependency tree as chain-of-custody evidence | CI/CD pipeline build artifacts and container image layer metadata (docker history) for any images published while axios 1.15.0 was in the dependency tree, to establish the blast radius of affected deployed artifacts across environments

Per-Action IR Details

Step 1: Containment — Identify all applications and services in your environment that depend on axios 1.15.0 by running 'npm list axios' or scanning your software bill of materials (SBOM). Temporarily restrict outbound proxy routing for any service confirmed to use this version until patched.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: isolate affected components to prevent further exposure of internal traffic through misconfigured NO_PROXY controls

Controls: NIST IR-4 (Incident Handling), NIST CM-8 (System Component Inventory), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: For teams without an SBOM platform, run 'find / -name package-lock.json -o -name package.json 2>/dev/null | xargs grep -l "axios"' across all app servers, then pipe through 'grep -A1 "\"axios\"" ' to extract pinned versions. For containers, run 'docker inspect --format="{{.Config.Labels}}" $(docker ps -q)' and cross-reference with 'docker run --rm npm list axios --depth=0'. Block outbound proxy traffic from affected hosts using iptables: 'iptables -I OUTPUT -p tcp --dport 3128 -j DROP' (adjust port for your proxy) as a temporary measure.

Evidence: Before restricting proxy routing, capture a full snapshot of current proxy access logs (Squid: /var/log/squid/access.log; nginx proxy: /var/log/nginx/access.log) filtered for requests sourced from 127.x.x.x addresses other than 127.0.0.1 — these are the forensic indicator of CVE-2026-42043 bypass activity. Also export the full npm dependency tree from each affected application host via 'npm list --json > npm-tree-$(hostname)-$(date +%Y%m%d).json' before any changes.

Step 2: Detection — Query your dependency inventory, CI/CD pipeline artifacts, and container image registries for axios@1.15.0. In proxy access logs, look for unexpected requests originating from loopback addresses in the 127.0.0.0/8 range that passed through your proxy rather than being blocked. Review Node.js application logs for HTTP requests targeting 127.x.x.x addresses other than 127.0.0.1.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: correlate proxy access logs and Node.js application logs to identify traffic that exploited the incomplete loopback subnet check in axios 1.15.0

Controls: NIST SI-4 (System Monitoring), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-2 (Event Logging), NIST AU-3 (Content of Audit Records), CIS 8.2 (Collect Audit Logs), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Without a SIEM, grep proxy access logs directly: 'grep -E "^127\.[1-9]|^127\.([1-9][0-9]|[1-9][0-9]{2})\." /var/log/squid/access.log | awk '{print $3, $7}' > loopback-proxy-hits.txt'. For CI/CD pipelines (GitHub Actions, GitLab CI), search pipeline artifacts: 'grep -r "axios" .github/workflows/ && grep -r "axios@1.15.0" Dockerfile*'. For container

registries without tooling, use Grype (free, Anchore): 'grype registry:: | grep axios' to identify vulnerable images without pulling them. Use osquery to query running Node.js processes: 'SELECT pid, name, cmdline FROM processes WHERE name LIKE "%node%";' then cross-reference PIDs with open network connections.

Evidence: Capture proxy access logs showing the full request URI, source IP, and upstream destination for any 127.0.0.0/8-sourced requests — this is the primary forensic indicator that the NO_PROXY bypass was triggered. Preserve Node.js application stdout/stderr logs (typically in /var/log// or journald: 'journalctl -u --since "72 hours ago" > app-log-$(date +%Y%m%d).txt') for HTTP outbound request records. Extract package-lock.json and node_modules/.package-lock.json from all application deployments as chain-of-custody artifacts proving the presence of axios 1.15.0 at detection time.

Step 3: Eradication — Upgrade axios to the latest version available on npm that explicitly addresses CVE-2026-42043 and GHSA-pmwg-cvhr-8vh7. Confirm the release notes or changelog reference the NO_PROXY loopback subnet fix before deploying. Run 'npm audit' post-upgrade to verify no residual advisories remain for axios.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: remove the vulnerable axios 1.15.0 instance from all affected environments and verify the fix explicitly addresses the full 127.0.0.0/8 subnet check, not just 127.0.0.1

Controls: NIST SI-2 (Flaw Remediation), NIST CM-3 (Configuration Change Control), NIST SA-10 (Developer Configuration Management), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Before upgrading in production, validate the candidate axios version in an isolated Node.js environment: 'npm install axios@ && node -e "const axios = require(\"axios\"); console.log(axios.VERSION);"'. Verify the changelog on the npm registry page or GitHub release notes at https://github.com/axios/axios/releases explicitly references GHSA-pmwg-cvhr-8vh7 or CVE-2026-42043 and mentions subnet-range validation (not just a point address fix). Run 'npm audit --json | jq ".vulnerabilities | keys"' post-upgrade to confirm axios advisories are cleared. For containerized apps, rebuild images from scratch rather than patching in-place to eliminate any residual node_modules state.

Evidence: Before executing the upgrade, preserve immutable snapshots of the current node_modules/axios/lib/helpers/isURLSameOrigin.js and node_modules/axios/lib/utils.js (or equivalent proxy-handling source files in axios 1.15.0) — these files contain the incomplete loopback check logic and constitute forensic evidence of the vulnerable code path. Hash them with 'sha256sum node_modules/axios/lib/helpers/*.js > axios-pre-patch-hashes.txt' for audit records per NIST AU-10 (Non-Repudiation).

Step 4: Recovery — After upgrading, validate that NO_PROXY rules correctly block requests to the full 127.0.0.0/8 range by running integration tests that target alternate loopback addresses. Monitor proxy logs for 48 hours post-remediation for any anomalous loopback-sourced traffic that bypasses NO_PROXY controls.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: verify that the patched axios version enforces NO_PROXY for the complete 127.0.0.0/8 subnet and confirm no loopback-sourced proxy traffic persists post-upgrade

Controls: NIST SI-6 (Security and Privacy Function Verification), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST CP-10 (System Recovery and Reconstitution), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Write a minimal Node.js integration test to confirm the fix: set NO_PROXY=127.0.0.0/8 in the test environment, then use the upgraded axios to attempt an HTTP request to 127.0.0.2 and 127.255.255.254 — the requests should fail or bypass the proxy, not route through it. Script: 'NO_PROXY="127.0.0.0/8" node -e "const axios=require(\"axios\"); axios.get(\"http://127.0.0.2:9999\").catch(e => console.log(\"Blocked as expected:\", e.code));"'. For 48-hour proxy log monitoring without a SIEM, run a cron job every 15 minutes: 'grep -cE "CONNECT 127\." /var/log/squid/access.log >> /tmp/loopback-count.log' and alert on any non-zero count.

Evidence: Capture post-upgrade proxy access logs under the same grep filter used in Step 2 to create a before/after comparison artifact. Preserve the 'npm list axios --json' output post-upgrade and diff it against the pre-patch snapshot to confirm version transition from 1.15.0. Document integration test results (pass/fail with timestamps) as evidence of

functional verification per NIST SI-6 (Security and Privacy Function Verification).

Step 5: Post-Incident — Review your patch management process for third-party npm dependencies, specifically the gap between incomplete-fix CVEs and full remediation. Evaluate whether your SBOM tooling flags incomplete fixes (CWE-184) as distinct from new vulnerabilities. Consider adding automated checks for the full loopback subnet in proxy configuration validation.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: conduct lessons-learned review focused on the detection gap created by CWE-184 incomplete-fix advisories for transitive npm dependencies like axios, and update scanning processes to treat GHSA incomplete-fix advisories as high-priority

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST SI-2 (Flaw Remediation), NIST SI-5 (Security Alerts, Advisories, and Directives), NIST RA-3 (Risk Assessment), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: Configure 'npm audit' in CI/CD pipelines to fail builds on HIGH or CRITICAL advisories, including GHSA entries: add '"scripts": {"audit-check": "npm audit --audit-level=high"}' to package.json and enforce it as a required pipeline step. For SBOM tooling without CWE-184 differentiation, use Grype with '--fail-on high' and cross-reference results against the GitHub Advisory Database (GHSA) manually for any axios advisories tagged as incomplete fixes. Add a proxy configuration test to your deployment pipeline that explicitly tests 127.0.0.2 through 127.255.255.254 connectivity to catch future NO_PROXY subnet-scope regressions before they reach production.

Evidence: Compile a post-incident artifact package: the pre-patch npm dependency trees, proxy log excerpts showing loopback-sourced requests (if any were observed), the axios version transition record, and integration test results. Document the timeline delta between the GHSA-pmwg-cvhr-8vh7 advisory publication and internal detection of axios 1.15.0 in your environment — this gap metric is the primary lessons-learned input for improving GHSA advisory monitoring per NIST SI-5 (Security Alerts, Advisories, and Directives).

Detection Guidance

Search your dependency manifests, lockfiles (package-lock.json, yarn.lock), and container images for 'axios' at version '1.15.0'. In Node.js runtime logs and proxy access logs, flag any HTTP requests routed through your proxy that originate from loopback addresses in the 127.0.0.0/8 range excluding 127.0.0.1 (i.e., 127.0.0.2 through 127.255.255.254); these should be blocked by NO_PROXY rules and their presence indicates potential bypass activity. No public IOCs (IPs, domains, hashes) are associated with active exploitation of this CVE at this time. EPSS score of 0.044% confirms low observed exploitation activity as of the configuration date.

Framework Mappings

MITRE-ATTACK

- **T1090** — Proxy
- **T1071.001** — Web Protocols

OWASP-TOP10-2021

- **A03:2021** — Injection

NIST-800-53R5

- **SI-10** — Information Input Validation

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

ISO-27001-2022

- **A.8.26** — Application security requirements
- **A.8.8** — Management of technical vulnerabilities

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1090	Proxy	Command-And-Control
T1071.001	Web Protocols	Command-And-Control

Sources

Source	URL	Tier
osv	https://osv.dev/vulnerability/GHSA-pmwg-cvhr-8vh7	T3
CVE-2026-42043 - CVE Record	https://www.cve.org/CVERecord?id=CVE-2026-42043	T3
Linux Distros Unpatched Vulnerability : CVE-2026-42043 - Tenable	https://www.tenable.com/plugins/nessus/310478	T3
CVE-2026-42043: Axios is a promise based HTTP client ... - Averlon	https://research.averlon.ai/vulnerability-intelligence/cve/CVE-2026...	T3
CVE-2026-42043 (High) detected in axios-1.15.0.tgz #11850 - GitHub	https://github.com/opensearch-project/OpenSearch-Dashboards/issues/...	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-42043	T1

Generated 2026-05-06 08:41 UTC by TJS Security Command Center