INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-25 18:37 UTC

DPRK Supply Chain Strike: Axios npm Compromise Exposes 70M+ Weekly Downloads to Cross-Platform Backdoor

THREAT CAMPAIGN | **HIGH** | CVSS 9.5

SCC Item ID	SCC-CAM-2026-0221
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	9.5
Affected Products	Axios npm package versions 1.14.1 and 0.30.4; Node.js ecosystem on Linux, macOS, and Windows; any organization consuming Axios via npm with 70M+ weekly downloads affected
Discovery Source	Rss:T1 Threatintel

Executive Summary

On March 31, 2026, a North Korean state-sponsored threat actor used stolen credentials to publish backdoored versions of Axios, one of the most widely used JavaScript libraries in the world, with over 70 million weekly downloads. Any organization whose development pipelines or production applications consumed Axios versions 1.14.1 or 0.30.4 via npm may have received a cross-platform backdoor without triggering standard integrity checks. The business risk is severe: compromised CI/CD pipelines can introduce malicious code into software products shipped to customers, exposing downstream customers to potential data compromise and breach notification obligations across the software supply chain.

Technical Analysis

Affected packages: Axios v1.14.1 and v0.30.4 published to the npm registry on March 31, 2026. The threat actor obtained npm maintainer credentials (T1078) and published trojanized releases that introduced a rogue dependency, 'plain-crypto-js', which is not a legitimate package and was created solely to deliver backdoor functionality. The backdoor operates cross-platform (Windows, macOS, Linux) and establishes JSON-based C2 over standard web protocols (T1071.001). Post-install scripts perform cleanup of package.json to remove evidence (T1070). Additional techniques include obfuscated payloads (T1027), JavaScript interpreter abuse (T1059.007), PowerShell execution on Windows (T1059.001), Unix shell execution on Linux/macOS (T1059.004), file and directory discovery (T1083), and ingress tool transfer (T1105). No CVE has been assigned. Applicable CWEs: CWE-506 (Embedded Malicious Code), CWE-494 (Download of Code Without Integrity Check), CWE-829 (Inclusion of Functionality from Untrusted Control Sphere). CVSS assessment:

Vendor security teams (CrowdStrike, Microsoft) assessed the backdoor payload severity at 9.5 (Critical); no official CVE-based CVSS is assigned. Attribution: STARDUST CHOLLIMA (CrowdStrike), UNC1069 (Google/Mandiant), Sapphire Sleet (Microsoft), assessed as overlapping DPRK-nexus clusters with financial motivation. MITRE ATT&CK primary technique: T1195.001 (Supply Chain Compromise: Compromise Software Dependencies and Development Tools).

Action Checklist

1. Containment: Immediately audit all package-lock.json, yarn.lock, and pnpm-lock.yaml files across development, CI/CD, and production environments for Axios v1.14.1 or v0.30.4. Block these versions at your artifact proxy (Artifactory, Nexus, AWS CodeArtifact) by adding them to blocked-versions or artifact rules, and prevent any new installs via policy enforcement. If either version is present in a running production environment, isolate the affected host or container pending investigation.
2. Detection: Search npm dependency trees and lockfiles for 'plain-crypto-js' as a direct or transitive dependency; its presence is a high-confidence indicator of compromise. Review CI/CD build logs from March 31, 2026 onward for npm install activity pulling Axios v1.14.1 or v0.30.4. Hunt for outbound JSON-formatted C2 traffic over HTTP/HTTPS from Node.js processes (T1071.001). On Windows, review PowerShell execution logs (Event ID 4104 for script block activity, Event ID 4688 for process creation) for activity spawned from Node.js parent processes. On Linux/macOS, review shell history and process audit logs (auditd, EDR) for shell commands invoked from node processes. Check file system for unexpected binaries or scripts dropped in temp directories post npm install.
3. Eradication: Remove Axios v1.14.1 and v0.30.4 from all environments. Upgrade to the latest verified clean release of Axios confirmed by the official maintainer via the npm registry advisory or the Axios GitHub repository. Remove 'plain-crypto-js' from all dependency trees. Rotate any secrets, tokens, or credentials that were accessible to processes running in environments where the backdoored package was installed. Rebuild all container images and deployment artifacts from a clean baseline.
4. Recovery: After upgrading Axios to a verified clean version, re-run full dependency audits using 'npm audit' and a software composition analysis (SCA) tool to confirm no residual malicious dependencies. Validate that 'plain-crypto-js' no longer appears in any dependency tree. Monitor network telemetry for continued C2 callbacks from affected hosts for at least 30 days. Confirm CI/CD pipelines are building from clean artifact sources and that lockfiles have been regenerated and committed.
5. Post-Incident: Implement npm package integrity controls: enforce 'npm ci' over 'npm install' in CI/CD pipelines to lock dependency resolution to lockfile hashes. Evaluate adoption of provenance attestation for npm packages (npm provenance, Sigstore). Implement SCA scanning as a mandatory pipeline gate. Establish a credential rotation policy for all package registry maintainer accounts and enforce MFA on npm accounts with publish access. Map this event to MITRE T1195.001 and assess whether your threat model adequately accounts for supply chain compromise via stolen maintainer credentials.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO, legal counsel, and initiate breach notification assessment immediately if forensic analysis confirms the backdoor was active in any environment with access to PII, PHI, payment card data, or credentials to production systems, or if C2 callback traffic is confirmed — given the DPRK state-actor attribution and 70M+ weekly download blast radius, assume credential exfiltration until forensically excluded.
Recovery Notes	After rebuilding from clean baselines, monitor all previously affected hosts and CI/CD runners for outbound C2 callbacks for a minimum of 30 days, as DPRK-attributed implants have historically used long beacon intervals and dormant activation mechanisms that survive simple package removal. Re-verify the integrity hash of every Axios version pinned in lockfiles against the npm registry and the official Axios GitHub release SHA after any future package update, and treat any hash mismatch as a new incident. Confirm that all rotated credentials — including npm publish tokens, CI/CD secrets, and application runtime credentials — have been fully invalidated at the issuing authority and not merely overwritten locally.
Forensic Artifacts	node_modules/plain-crypto-js/ directory contents (package.json, index.js, any .node compiled binaries) — the malicious payload dropped by the backdoored Axios package; SHA-256 hash all files before removal | package-lock.json integrity hash field for axios entries — the resolved _integrity SHA-512 value for Axios v1.14.1 or v0.30.4 will differ from the legitimate hash published by the Axios maintainers, providing direct cryptographic evidence of package substitution at the npm registry level | CI/CD build logs from March 31, 2026 onward showing npm install resolution output — preserves the registry endpoint, resolved version, and install timestamp establishing which pipelines ingested the backdoored package and the precise exposure window | Network flow records or pcap of outbound HTTP/HTTPS POST traffic from node.exe or node processes to external IPs, specifically JSON-formatted request bodies consistent with MITRE T1071.001 C2 beaconing from the plain-crypto-js backdoor component | Process tree logs from Sysmon Event ID 1 (Windows) or auditd EXECVE syscall records (Linux) showing node.exe or node as a parent process of cmd.exe, powershell.exe, sh, or bash — indicative of the backdoor's command execution capability being triggered post-install

Per-Action IR Details

Containment — Immediately audit all package-lock.json, yarn.lock, and pnpm-lock.yaml files across development, CI/CD, and production environments for Axios v1.14.1 or v0.30.4. Block these versions at your artifact proxy (Artifactory, Nexus, AWS CodeArtifact) and prevent any new installs. If either version is present in a running production environment, isolate the affected host or container pending investigation.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy

Controls: NIST IR-4 (Incident Handling), NIST CM-3 (Configuration Change Control), NIST SI-7 (Software, Firmware, and Information Integrity), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.3 (Address Unauthorized Software)

Compensating: Run 'find . -name package-lock.json -o -name yarn.lock -o -name pnpm-lock.yaml | xargs grep -l "axios.*1\.14\.1\|axios.*0\.30\.4"' recursively across all repo roots and CI workspace directories. For artifact proxy blocking without enterprise tooling, add a .npmrc deny-list entry or use 'npm pack' interception via a local Verdaccio proxy instance configured to reject those specific version strings. For container isolation without an orchestrator, use 'docker network disconnect' on the running container and flag it for snapshot before teardown.

Evidence: Before isolating any host or container, capture: (1) full output of 'npm list --all --json' from the affected project root to preserve the resolved dependency tree at time of incident; (2) a filesystem snapshot or 'docker commit' of any running container that installed the backdoored Axios version to preserve the post-install state including any dropped files; (3) contents of all lockfiles verbatim — the resolved integrity hash field for axios in package-lock.json will differ from the legitimate SHA-512 published by the Axios maintainers for the same version string, which is direct evidence of package substitution.

Detection — Search npm dependency trees and lockfiles for 'plain-crypto-js' as a direct or transitive dependency; its presence is a high-confidence indicator of compromise. Review CI/CD build logs from March 31, 2026 onward for npm install activity pulling Axios v1.14.1 or v0.30.4. Hunt for outbound JSON-formatted C2 traffic over HTTP/HTTPS from Node.js processes (T1071.001). On Windows, review PowerShell execution logs (Event ID 4104) for activity spawned from Node.js parent processes. On Linux/macOS, review shell history and process audit logs (auditd, EDR) for shell commands invoked from node processes. Check file system for unexpected binaries or scripts dropped in temp directories post npm install.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST SI-4 (System Monitoring), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: IOC sweep for 'plain-crypto-js': run 'find /path/to/repos -name node_modules -prune -o -name package-lock.json -print | xargs grep -l plain-crypto-js' across all project roots. For C2 traffic detection without a SIEM, use 'tcpdump -i any -w axios-c2-capture.pcap' on affected hosts and filter in Wireshark for HTTP/HTTPS POST streams from node processes with JSON payloads to non-inventory external IPs. On Linux, use 'auditctl -a always,exit -F arch=b64 -S execve -F ppid=$(pgrep node) -k node_child_exec' to capture child processes spawned by node. On Windows without EDR, deploy Sysmon with a config that logs Event ID 1 (Process Create) filtering on ParentImage containing 'node.exe' and Event ID 3 (Network Connect) from node.exe to external IPs. CI/CD log review: grep build logs for 'added.*axios' or 'resolved.*axios@1.14.1' or 'axios@0.30.4' in npm install output from March 31, 2026 onward.

Evidence: Preserve before analysis: (1) CI/CD build log archives from March 31, 2026 onward — specifically npm install stdout/stderr showing resolved package versions and registry URLs, to establish which pipelines pulled the backdoored package and from which registry endpoint; (2) contents of node_modules/plain-crypto-js/ directory if present, including package.json, index.js, and any compiled .node binaries, which constitute the malicious payload artifact; (3) on Windows, export PowerShell Script Block Logging events (Event ID 4104) from the Microsoft-Windows-PowerShell/Operational log filtered to the timeframe post-install; (4) on Linux/macOS, /tmp and $TMPDIR directory listings timestamped to the npm install window, and ~/.npm/_logs/ for npm debug logs showing install resolution chain; (5) network flow records or pcap from affected hosts showing outbound connections from node.exe/node process to external IPs, specifically JSON-body POST requests consistent with MITRE T1071.001 application-layer C2.

Eradication — Remove Axios v1.14.1 and v0.30.4 from all environments. Upgrade to the latest verified clean release of Axios confirmed by the official maintainer via the npm registry advisory or the Axios GitHub repository. Remove 'plain-crypto-js' from all dependency trees. Rotate any secrets, tokens, or credentials that were accessible to processes running in environments where the backdoored package was installed. Rebuild all container images and deployment artifacts from a clean baseline.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-3 (Malicious Code Protection), NIST IA-5 (Authenticator Management), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 7.4 (Perform Automated Application Patch Management), CIS 5.2 (Use Unique Passwords)

Compensating: Verify the clean Axios release integrity before deploying: run 'npm view axios dist-tags.latest' and cross-check the published shasum against the official Axios GitHub release tag and the npm registry page audit advisory. For credential rotation scoping without a secrets manager, enumerate all .env files, CI/CD environment variable stores (GitHub Actions secrets, GitLab CI variables, Jenkins credentials store), and application config files accessible to the Node.js process at runtime — treat all as compromised. Rebuild container images by invalidating the Docker layer cache ('docker build --no-cache') and pinning the clean Axios version with an exact version string and verified integrity hash in package.json. Use 'npm ci --ignore-scripts' on the clean baseline to prevent post-install script execution during rebuild.

Evidence: Before wiping affected environments: (1) preserve a full image or snapshot of any host or container where plain-crypto-js was present — this is primary forensic evidence of the DPRK-attributed payload and may be required if regulatory notification is triggered; (2) document the complete list of environment variables, mounted secrets, and credential files in scope for rotation by reviewing the process environment of the node process ('cat /proc//environ' on Linux, or 'Get-Process node | Select-Object -ExpandProperty StartInfo' on Windows) — this scopes the credential rotation to what was actually exposed rather than rotating everything blindly; (3) record the SHA-256 hash of the malicious plain-crypto-js package files before removal to support threat intelligence sharing and future YARA/hash-based detection.

Recovery — After upgrading Axios to a verified clean version, re-run full dependency audits using 'npm audit' and a software composition analysis (SCA) tool to confirm no residual malicious dependencies. Validate that 'plain-crypto-js' no longer appears in any dependency tree. Monitor network telemetry for continued C2 callbacks from affected hosts for at least 30 days. Confirm CI/CD pipelines are building from clean artifact sources and that lockfiles have been regenerated and committed.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-6 (Audit Record Review, Analysis, and Reporting), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.3 (Perform Automated Operating System Patch Management)

Compensating: Run 'npm audit --audit-level=high' against the clean dependency tree and pipe output to a file for documentation. For SCA without a commercial tool, use the free tier of OWASP Dependency-Check or Grype (anchore/grype) against the rebuilt node_modules directory — both support Node.js ecosystems and will flag known malicious packages. For ongoing C2 callback monitoring without a SIEM, configure a cron job on a network boundary device or use Zeek (formerly Bro) with a custom notice script alerting on outbound HTTP/HTTPS POST connections from the recovered host IP range to IPs not in your approved egress inventory. Commit regenerated lockfiles to version control with a signed commit and require peer review before merge.

Evidence: At recovery validation: (1) run 'npm list plain-crypto-js --all' on every rebuilt environment and retain the output showing the package is absent as evidence of successful eradication; (2) capture a fresh 'npm list --all --json' output from the clean environment and compare the integrity hashes in the regenerated package-lock.json against npm registry published hashes for Axios to confirm clean resolution; (3) retain 30-day network flow exports showing no outbound connections from recovered hosts to the C2 infrastructure identified during the detection phase — this 30-day window accounts for the possibility that the backdoor included dormant callback timers or beacon jitter common in DPRK-attributed implants.

Post-Incident — Implement npm package integrity controls: enforce 'npm ci' over 'npm install' in CI/CD pipelines to lock dependency resolution to lockfile hashes. Evaluate adoption of provenance attestation for npm packages (npm provenance, Sigstore). Implement SCA scanning as a mandatory pipeline gate. Establish a credential rotation policy for all package registry maintainer accounts and enforce MFA on npm accounts with publish access. Map this event to MITRE T1195.001 and assess whether your threat model adequately accounts for supply chain compromise via stolen maintainer credentials.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST SI-7 (Software, Firmware, and Information Integrity), NIST SA-12 (Supply Chain Protection), NIST IA-5 (Authenticator Management), CIS 6.3 (Require MFA for Externally-Exposed Applications), CIS 6.5 (Require MFA for Administrative Access), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: For teams without a commercial SCA platform: integrate Grype or OWASP Dependency-Check as a free CI/CD pipeline gate by adding a pipeline step that fails the build if high-severity findings are detected — both tools have GitHub Actions and GitLab CI integrations. For npm provenance without enterprise PKI, enable npm's built-in provenance attestation (available free for public packages on npmjs.com) which ties published packages to a verifiable GitHub Actions workflow run via Sigstore transparency logs. Write a Sigma rule targeting Sysmon Event ID 1 for

node.exe spawning cmd.exe, powershell.exe, or sh as child processes to detect future backdoor activation in CI/CD runners. Document the MITRE T1195.001 (Supply Chain Compromise: Compromise Software Dependencies and Development Tools) mapping in your threat model and add a specific detection hypothesis for stolen package maintainer credentials to your threat hunting program.

Evidence: For the post-incident review: (1) reconstruct the full attack timeline from CI/CD build logs showing first introduction of Axios v1.14.1 or v0.30.4 into any pipeline, through to first detected C2 callback or IOC match, to establish dwell time and blast radius; (2) inventory all npm packages in your dependency trees where your organization has no visibility into maintainer account security posture — this scopes future supply chain risk assessment; (3) document all credentials confirmed or suspected to have been in scope of exposure during the backdoor's active window, with rotation timestamps, for the lessons-learned record and any required regulatory notification package.

Detection Guidance

Primary IOC: presence of 'plain-crypto-js' in package-lock.json, yarn.lock, or pnpm-lock.yaml as a direct or transitive dependency of Axios. Query artifact repositories and file systems for this package name. Version-specific indicators: Axios v1.14.1 or v0.30.4 in any lockfile or installed node_modules directory. Behavioral indicators: Node.js processes making outbound HTTP/HTTPS requests with JSON-structured payloads to unfamiliar external hosts (C2 pattern per T1071.001); PowerShell processes (Event ID 4104 for script block activity, Event ID 4688 for process creation) with Node.js as parent process (Windows); unexpected shell execution from node processes (auditd event type EXECVE on Linux). Post-install artifact: review npm postinstall script execution logs for cleanup activity targeting package.json (T1070). EDR telemetry: look for file writes to temp or staging directories immediately following npm install operations in CI/CD pipeline runners. If you have a SIEM, query for process trees: node.exe or node spawning cmd.exe, powershell.exe, /bin/sh, or /bin/bash within 60 seconds of a package install event. Source attribution context: CrowdStrike, Microsoft, and Trend Micro have published threat intelligence on this campaign; cross-reference their published IOC sets against your environment.

Indicators of Compromise

Type	Value	Context	Confidence
DOMAIN	`plain-crypto-js (npm package name)`	Rogue npm package introduced as a dependency by trojanized Axios v1.14.1 and v0.30.4; not a legitimate package; presence in any dependency tree is a high-confidence indicator of compromise	HIGH
HASH	`[not published in available sources — check CrowdStrike and Microsoft advisories for package hash IOCs]`	npm package integrity hashes for Axios v1.14.1 and v0.30.4; verify against published IOC sets from CrowdStrike and Microsoft	HIGH

Framework Mappings

MITRE-ATTACK

- **T1105** — Ingress Tool Transfer

- **T1140** — Deobfuscate/Decode Files or Information
- **T1059.007** — JavaScript
- **T1078** — Valid Accounts
- **T1059.001** — PowerShell
- **T1071.001** — Web Protocols
- **T1027** — Obfuscated Files or Information
- **T1071** — Application Layer Protocol
- **T1070** — Indicator Removal
- **T1059.004** — Unix Shell
- **T1083** — File and Directory Discovery
- **T1195.001** — Compromise Software Dependencies and Development Tools
- **T1204.002** — Malicious File

NIST-800-53R5

- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CM-7** — Least Functionality
- **SI-7** — Software, Firmware, and Information Integrity
- **CM-3** — Configuration Change Control
- **SR-2** — Supply Chain Risk Management Plan

OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures

CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **6.3** — Require MFA for Externally-Exposed Applications
- **15.1** — Establish and Maintain an Inventory of Service Providers

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

ISO-27001-2022

- **A.5.34** — Privacy and protection of personal information
- **A.5.21** — Managing information security in the ICT supply chain

NIST-CSF-2

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1105	Ingress Tool Transfer	Command-And-Control
T1140	Deobfuscate/Decode Files or Information	Defense-Evasion
T1059.007	JavaScript	Execution
T1078	Valid Accounts	Defense-Evasion
T1059.001	PowerShell	Execution
T1071.001	Web Protocols	Command-And-Control
T1027	Obfuscated Files or Information	Defense-Evasion
T1071	Application Layer Protocol	Command-And-Control
T1070	Indicator Removal	Defense-Evasion
T1059.004	Unix Shell	Execution
T1083	File and Directory Discovery	Discovery
T1195.001	Compromise Software Dependencies and Development Tools	Initial-Access
T1204.002	Malicious File	Execution

Sources

Source	URL	Tier
Blog	https://www.crowdstrike.com/en-us/blog/stardust-chollima-likely-com...	T3
	https://thehackernews.com/2026/04/google-attributes-axios-npm-suppl...	T3
	https://backendnews.net/crowdstrike-stolen-credentials-used-in-axio...	T3
	https://www.microsoft.com/en-us/security/blog/2026/04/01/mitigating...	T1

Source	URL	Tier
Axios NPM Package Compromised: Supply Chain Attack Hits ...	https://www.trendmicro.com/en_us/research/26/c/axios-npm-package-co...	T3

Generated 2026-04-25 18:37 UTC by TJS Security Command Center