INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-14 13:36 UTC

China and DPRK Lead Multi-Front Campaign Against Technology Sector: Supply Chain, Insider Threats, and IP Theft Drive 2025-2026 Targeting Surge

THREAT CAMPAIGN | **HIGH** | CVSS 9.5

SCC Item ID	SCC-CAM-2026-0458
Type	Threat Campaign
Severity	HIGH
CVSS Base Score	9.5
Affected Products	GitHub repositories, npm ecosystem (Axios package v1.14.1 and v0.30.4), macOS platforms, North American and European technology organizations, mail infrastructure
Discovery Source	Rss:T1 Threatintel

Executive Summary

China-nexus and North Korean state-sponsored threat actors conducted sustained, coordinated campaigns against technology organizations from April 2025 through March 2026, targeting intellectual property, AI research, and developer supply chains. The most acute risk is the confirmed compromise of the Axios npm package (versions v1.14.1 and v0.30.4), which planted a remote access trojan inside one of the most widely deployed JavaScript libraries, exposing any organization that updated without integrity verification. Compounding state-actor activity, elevated access broker operations and ransomware activity in the technology sector signal a sustained threat environment requiring immediate supply chain audit and insider threat review.

Technical Analysis

This campaign encompasses two primary threat tracks. First, China-nexus actors pursued IP theft and AI capability acquisition against North American and European technology organizations using techniques including credential access (T1110.003, password spraying), valid account abuse (T1078), command execution (T1059), and data exfiltration to code repositories (T1567.001). Second, DPRK actors are assessed to have combined fraudulent IT worker insertion (T1136, T1078) with a supply chain compromise of the Axios npm package. Malicious versions v1.14.1 and v0.30.4 were published to the npm registry and included a remote access trojan, exploiting CWE-494 (Download of Code Without Integrity Check) and CWE-829 (Inclusion of Functionality from Untrusted Control Sphere). CWE-798 (Use of Hard-coded Credentials) is present in RAT

payloads; CWE-287 (Improper Authentication) applies to insider threat vector through weak contractor identity vetting. MITRE techniques span initial access through supply chain compromise (T1195.001, T1195.002), resource development (T1583, T1588.001), phishing (T1566), exploitation of public-facing applications (T1190), data collection from information repositories (T1213), and financial extortion (T1657). No CVE ID is associated with the Axios compromise; the attack vector was registry account compromise enabling malicious package publication. Affected platforms include macOS and any Node.js or JavaScript environment that consumed the poisoned versions. Organizations should consult the official npm security advisory for the verified clean release version and deployment timeline.

Action Checklist

1. Step 1: Containment. Immediately audit all Node.js and JavaScript projects for Axios dependency versions v1.14.1 and v0.30.4. Block installation of these versions at the package manager or artifact proxy level (Nexus, Artifactory, Verdaccio). Isolate any build pipelines, CI/CD agents, or production services confirmed to have consumed either version. Per NIST SP 800-53 control AC-2 (Account Management) and CIS 2.1 (Establish and Maintain a Software Inventory), enumerate all software consuming Axios across the enterprise before scoping further.
2. Step 2: Detection. Query SIEM and EDR for processes spawned from Node.js runtimes on unexpected outbound connections, particularly to non-standard ports or newly registered domains, consistent with RAT beacon behavior (T1059, T1583). Review npm audit logs and package-lock.json or yarn.lock files in all repositories for Axios v1.14.1 or v0.30.4. Check CI/CD pipeline execution logs for the period the malicious packages were available. Apply NIST AU-6 (Audit Record Review, Analysis, and Reporting): review logs for signs of lateral movement or credential access following any confirmed consumption of the poisoned package. Look for anomalous account creation events (T1136) and use of valid but unexpected accounts (T1078). If RAT callback activity is confirmed, escalate to incident response immediately.
3. Step 3: Eradication. Upgrade Axios to the safe version indicated in the official npm security advisory (consult npm registry advisory at remediation time). Remove the malicious versions from any internal artifact registries. Rotate all credentials, API keys, and tokens accessible from any environment that executed the compromised package, addressing CWE-798 (Hard-coded Credentials) and CWE-287 (Improper Authentication) exposure. Apply credential rotation and hardening practices per NIST SP 800-53 SC-3 (Security Posture Updates). Review and revoke any accounts created or modified during the exposure window per AC-2 (Account Management) and AC-6 (Least Privilege).
4. Step 4: Recovery. After upgrading, re-run full dependency audits (npm audit, Dependabot, or equivalent) across all repositories to confirm no residual malicious packages. Validate that all build artifacts produced during the exposure window are untrusted and should be rebuilt from clean source. Monitor post-remediation network telemetry for continued RAT beacon activity, which would indicate persistence mechanisms survived package removal. Apply NIST SI-4 (Information System Monitoring) and AU-12 (Audit Record Generation) to ensure logging coverage is in place for affected systems. Confirm MFA is enforced on all npm publishing accounts and code repository administrative access per CIS 6.3 and CIS 6.5.
5. Step 5: Post-Incident. Conduct a formal supply chain risk review against NIST SP 800-161r1 principles, focusing on third-party package integrity verification. Implement or enforce Subresource Integrity (SRI) checks and package signature verification to address CWE-494 structurally. Establish a process for monitoring npm account compromise alerts and registry anomalies. Review insider threat controls in light of DPRK IT worker activity: verify identity vetting for remote contractors, enforce separation of duties per

AC-5, and restrict access to sensitive repositories to need-to-know per AC-3 (Access Enforcement) and AC-6 (Least Privilege). Review access broker exposure by auditing dark web monitoring feeds for corporate credentials.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to CISO, legal counsel, and external IR retainer immediately if: any confirmed execution of the Axios RAT payload is detected in production environments; credential rotation reveals secrets were accessed post-compromise; any DPRK IT worker insider threat is identified among contractor staff; or if affected systems process PII, PHI, or export-controlled data triggering breach notification obligations under GDPR, CCPA, HIPAA, or EAR/ITAR.
Recovery Notes	All build artifacts — container images, compiled binaries, deployment packages — produced during the Axios v1.14.1 or v0.30.4 exposure window must be treated as untrusted and rebuilt from verified-clean source with a confirmed-safe Axios version before re-deployment to production; do not attempt to patch artifacts in place. Monitor network egress from formerly affected CI/CD agents and production Node.js services for at least 30 days post-remediation using DNS and NetFlow telemetry, as the RAT may have established persistence mechanisms (scheduled tasks, cron jobs, or injected npm lifecycle scripts) that survive the package upgrade. Validate integrity of the npm publishing accounts for any internal packages by reviewing the npm access audit log for unauthorized publish events that may indicate the threat actor attempted to pivot from the compromised Axios dependency to packages your organization maintains.
Forensic Artifacts	npm install logs and package-lock.json / yarn.lock files from all CI/CD pipelines during the exposure window — these establish which pipelines consumed Axios v1.14.1 or v0.30.4 and provide the precise timestamp of malicious code execution, which is the anchor for all downstream lateral movement analysis | Memory dump of Node.js processes that loaded the malicious Axios version — the RAT payload may have staged C2 configuration, exfiltrated credentials, or written in-memory persistence artifacts that exist only in the process heap and are destroyed on process termination or package upgrade | DNS query logs and NetFlow/IPFIX records from CI/CD agents and production Node.js services for the exposure window — the Axios RAT's beacon behavior would produce characteristic outbound connection patterns to newly registered or low-reputation domains on non-standard ports, distinguishable from normal npm registry and CDN traffic | GitHub / GitLab / Bitbucket audit logs covering the exposure window — specifically, events for repository access, package-lock.json commits, and any new account creations or permission grants, which would surface both the supply chain injection vector and any DPRK IT worker insider account activity during the same period | CI/CD environment variable snapshots and secrets manager access logs — the malicious Axios code executing inside a build pipeline had access to all secrets injected as environment variables at install time; access logs from AWS Secrets Manager, HashiCorp Vault, or GitHub Actions encrypted secrets reveal whether those credentials were subsequently used from unexpected sources, confirming exfiltration

Per-Action IR Details

Step 1: Containment — Immediately audit all Node.js and JavaScript projects for Axios dependency versions v1.14.1 and v0.30.4. Block installation of these versions at the package manager or artifact proxy level (Nexus, Artifactory, Verdaccio). Isolate any build pipelines, CI/CD agents, or production services confirmed to have

consumed either version. Per NIST IR control guidance and CIS 2.1 (Establish and Maintain a Software Inventory), enumerate all software consuming Axios across the enterprise before scoping further.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy: stop the bleeding by preventing further execution of the malicious Axios RAT payload across build and production environments before scoping eradication.

Controls: CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.3 (Address Unauthorized Software), NIST AC-4 (Information Flow Enforcement)

Compensating: Run `grep -r 'axios' /path/to/repos --include='package*.json' --include='yarn.lock' | grep -E '1\.14\.1|0\.30\.4'` across all source repositories. In Verdaccio or a self-hosted registry, add a deny rule for `axios@1.14.1` and `axios@0.30.4` in the package blocklist config. For CI agents without a proxy, add a pre-install npm hook (`preinstall` script in root `package.json`) that aborts if either version is detected.

Evidence: Before isolating any CI/CD agent or build server, capture: (1) running process list and parent-child process tree (`ps auxf` on Linux, `Get-CimInstance Win32_Process` on Windows) to identify any Node.js child processes spawned from the build agent; (2) active outbound network connections from the build agent (`ss -tunp` or `netstat -ano`) to capture any live RAT beacon channels the malicious Axios code may have opened; (3) environment variables (`/proc/self/environ` or `[Environment]::GetEnvironmentVariables()`) since the RAT may have exfiltrated secrets already present in the CI environment at install time. Volatile network state is destroyed the moment the agent is isolated.

Step 2: Detection — Query SIEM and EDR for processes spawned from Node.js runtimes on unexpected outbound connections, particularly to non-standard ports or newly registered domains, consistent with RAT beacon behavior (T1059, T1583). Review npm audit logs and package-lock.json or yarn.lock files in all repositories for Axios v1.14.1 or v0.30.4. Check CI/CD pipeline execution logs for the period the malicious packages were available. Apply NIST AU-6 (Audit Record Review, Analysis, and Reporting) discipline: review logs for signs of lateral movement or credential access following any confirmed consumption of the poisoned package. Look for anomalous account creation events (T1136) and use of valid but unexpected accounts (T1078), which DPRK IT worker programs rely on.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: correlate package consumption timelines with anomalous process and network behavior to determine which environments executed the Axios RAT payload and whether lateral movement or credential theft followed.

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-3 (Content Of Audit Records), CIS 8.2 (Collect Audit Logs)

Compensating: Without SIEM/EDR: (1) On Linux build agents, run `ausearch -c node --start today` or parse `/var/log/auth.log` for `node` process entries with outbound connections; (2) deploy Sysmon on Windows CI agents using the SwiftOnSecurity Sysmon config — Event ID 3 (Network Connection) filtered on `node.exe` as the initiating process reveals RAT beaconing; (3) use `jq` to parse all `package-lock.json` files: `find . -name 'package-lock.json' | xargs jq '.packages["node_modules/axios"].version // .dependencies.axios.version' 2>/dev/null`; (4) for account anomaly detection without SIEM, query Windows Security Event Log manually: `Get-WinEvent -FilterHashtable @{LogName='Security'; Id=4720,4728,4732} | Where-Object {$_.TimeCreated -gt (Get-Date).AddDays(-90)}` covering the full exposure window.

Evidence: Preserve before any account lockout or network block action: (1) full npm install/execution logs from CI pipelines during the exposure window (GitHub Actions logs, Jenkins build console output, GitLab CI job traces) — these establish the install timestamp and execution context of the malicious package; (2) DNS query logs from build agents for the RAT's C2 domain (check `/var/log/syslog` for systemd-resolved queries or Windows DNS client event log Event ID 3008); (3) `~/.npm/_logs/` directory on any developer workstation that ran `npm install` during the window, which contains timestamped install records; (4) Git commit history and `git log --all --oneline` for any repository where `package-lock.json` was modified to introduce the malicious version, which may reveal the injection vector or a compromised contributor account.

Step 3: Eradication — Upgrade Axios to a verified clean version per the npm registry advisory (v1.7.9 is the recommended v1.x safe release; confirm the specific safe version against the npm advisory at time of remediation). Remove the malicious versions from any internal artifact registries. Rotate all credentials, API keys, and tokens accessible from any environment that executed the compromised package, addressing CWE-798 (Hard-coded Credentials) and CWE-287 (Improper Authentication) exposure. Apply D3-CRO (Credential Rotation) and D3-CH (Credential Hardening) countermeasures. Review and revoke any accounts created or modified during the exposure window per AC-2 (Account Management) and AC-6 (Least Privilege).

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication: remove the malicious Axios versions from all artifact registries and dependency trees, then rotate every credential the RAT payload could have accessed from the CI/CD environment's memory or environment variables during execution.

Controls: NIST AC-2 (Account Management), NIST AC-6 (Least Privilege), CIS 5.1 (Establish and Maintain an Inventory of Accounts), CIS 6.2 (Establish an Access Revoking Process)

Compensating: Without automated secrets management: (1) enumerate all secrets exposed in CI by running `env | grep -iE 'key|token|secret|pass|api|auth'` on each affected build agent before shutdown and cross-reference against your secrets inventory; (2) use the GitHub audit log API (`GET /orgs/{org}/audit-log?phrase=axios`) to identify any repository that triggered a workflow consuming the malicious version; (3) for Nexus/Artifactory, manually delete the component via the repository manager UI or REST API (`curl -u admin:pass -X DELETE 'https://nexus/service/rest/v1/components/{id}'`); (4) use `npm deprecate axios@1.14.1 'MALICIOUS — do not install'` on any internal Verdaccio instance to add a blocking deprecation notice.

Evidence: Before rotating any credential or revoking any account, capture: (1) a memory dump of any long-running Node.js process (production service, not just build agent) that loaded Axios v1.14.1 or v0.30.4 — use `gcore ` on Linux or ProcDump on Windows — because the RAT may have staged credentials or C2 configuration in the heap that disappears on process termination; (2) a snapshot of all active sessions and OAuth tokens currently issued to service accounts associated with affected CI pipelines (GitHub Apps token list, AWS STS `get-caller-identity` output, GCP `gcloud auth list`) before they are rotated, to establish the blast radius of what the RAT could have accessed; (3) the full process execution history from the compromised environment (`/var/log/audit/audit.log` execve syscalls or Windows Security Event ID 4688) covering the install-to-detection window.

Step 4: Recovery — After upgrading, re-run full dependency audits (npm audit, Dependabot, or equivalent) across all repositories to confirm no residual malicious packages. Validate that all build artifacts produced during the exposure window are untrusted and should be rebuilt from clean source. Monitor post-remediation network telemetry for continued RAT beacon activity, which would indicate persistence mechanisms survived package removal. Apply NIST SI-4 equivalent monitoring discipline and AU-12 (Audit Record Generation) to ensure logging coverage is in place for affected systems. Confirm MFA is enforced on all npm publishing accounts and code repository administrative access per CIS 6.3 and CIS 6.5.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery: restore systems to verified-clean state by rebuilding all artifacts from trusted source, then validate that no RAT persistence survived package removal through sustained network and process monitoring.

Controls: NIST AU-12 (Audit Record Generation), NIST AU-2 (Event Logging), CIS 6.3 (Require MFA for Externally-Exposed Applications), CIS 6.5 (Require MFA for Administrative Access), CIS 7.4 (Perform Automated Application Patch Management)

Compensating: Without Dependabot or enterprise SCA tooling: (1) run `npm audit --json | jq '.vulnerabilities | keys[]'` across all repos in a shell loop and pipe results to a report file; (2) use YARA rules targeting the known malicious Axios payload signatures (if published by the npm security team or researchers) against local `node_modules/.cache` and build artifact archives; (3) for post-remediation beacon detection without EDR, configure Sysmon Event ID 3 alerts for `node.exe` network connections to any domain registered within the past 90 days, cross-referenced against a free WHOIS API or DomainTools Community edition; (4) verify npm MFA status for all package-publishing accounts via `npm profile get` and enforce with `npm profile set tfa auth-and-writes`.

Evidence: After package removal but before declaring recovery complete, collect: (1) a fresh `netstat -ano` or `ss -tunp` baseline from all previously affected hosts to compare against the pre-remediation snapshot and confirm no residual C2 beacon channels remain open; (2) file system integrity check output (`aide --check` on Linux or Windows SFC `/scannow` plus manual hash verification of rebuilt binaries) to confirm build artifacts produced from clean source differ from exposure-window artifacts at the hash level; (3) post-rotation authentication logs from npm registry and GitHub/GitLab (audit log API) for 30 days to detect any continued use of credentials that may have been exfiltrated by the RAT prior to rotation.

Step 5: Post-Incident — Conduct a formal supply chain risk review against NIST SP 800-161r1 principles, focusing on third-party package integrity verification. Implement or enforce Subresource Integrity (SRI) checks and package signature verification to address CWE-494 structurally. Establish a process for monitoring npm account compromise alerts and registry anomalies. Review insider threat controls in light of DPRK IT worker activity: verify identity vetting for remote contractors, enforce separation of duties per AC-5, and restrict access to sensitive repositories to need-to-know per AC-3 (Access Enforcement) and AC-6 (Least Privilege). Review access broker exposure by auditing dark web monitoring feeds for corporate credentials.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: conduct lessons-learned focused on supply chain integrity controls and insider threat vetting gaps exploited by DPRK IT worker programs, then operationalize detection improvements to prevent recurrence.

Controls: NIST AC-3 (Access Enforcement), NIST AC-5 (Separation Of Duties), NIST AC-6 (Least Privilege), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Without a commercial dark web monitoring subscription: (1) configure free Have I Been Pwned API alerts for all corporate email domains used by developers and contractor accounts; (2) implement npm package signature verification in CI using `npm audit signatures` (available in npm CLI v9+) to detect unsigned or signature-mismatched packages before install; (3) for DPRK IT worker insider threat detection without a UBA platform, create a manual quarterly review checklist: cross-reference contractor onboarding records against public breach databases, verify video-call identity against government ID, and audit VPN/access logs for geographic inconsistencies (logins from unexpected countries for a purportedly domestic contractor); (4) write a Sigma rule targeting `npm install` events in CI logs that resolve to packages with version strings matching the pattern of this incident (newly published patch versions of high-popularity packages with no corresponding GitHub release or changelog entry).

Evidence: For the post-incident record and to support threat intelligence sharing: (1) preserve sanitized copies of all `package-lock.json` and `yarn.lock` files that contained the malicious Axios versions as documentary evidence of the exposure scope; (2) archive all npm install logs, CI pipeline execution records, and network connection logs from the exposure window in write-once storage per NIST AU-11 (Audit Record Retention) requirements, as these may be required for regulatory notification or law enforcement referral given nation-state attribution; (3) document all contractor accounts active during the exposure window with their onboarding documentation, access scope, and activity logs to support any DPRK IT worker investigation referral to relevant authorities (CISA, FBI, or equivalent national authority).

Detection Guidance

Primary detection focus has two tracks. Track 1, Axios Supply Chain: Search all package-lock.json, yarn.lock, and requirements manifests in source control for Axios versions v1.14.1 or v0.30.4. In CI/CD logs, query for npm install events that resolved either version during the exposure window. In EDR telemetry, hunt for Node.js processes establishing outbound TCP connections to unfamiliar external hosts, especially on non-standard ports, which is consistent with RAT callback behavior. On macOS endpoints, review LaunchAgent and LaunchDaemon persistence locations for entries created by Node.js processes (D3-SICA, System Init Config Analysis). Use D3-SFA (System File Analysis) to check for modifications to node_modules directories outside of expected package manager operations. Track 2, State Actor TTPs: For DPRK IT worker insertion, look for new

accounts created without corresponding HR onboarding records (T1136), accounts accessing sensitive code repositories or IP stores outside business hours, and accounts with anomalous geographic login patterns (D3-LAM, Local Account Monitoring, NIST AU-6). For China-nexus IP theft, monitor for bulk file access or staging of large data volumes in cloud storage or code repositories followed by exfiltration to external git hosting (T1213, T1567.001). Alert on password spray patterns against Entra ID, Okta, or VPN infrastructure (T1110.003), look for high volumes of 401/403 responses across multiple accounts from a single source IP within a short window. NIST AU-2 (Event Logging) should cover authentication failures, account creation, and data access events as minimum log sources.

Indicators of Compromise

Type	Value	Context	Confidence
HASH	`see upstream advisories from StepSecurity and Trend Micro for package hashes of Axios v1.14.1 and v0.30.4`	Malicious npm package versions containing embedded RAT; specific file hashes should be pulled from the StepSecurity and Trend Micro advisories linked in sources, as hash values were not embedded in the provided source data	HIGH
URL	`https://www.npmjs.com/package/axios — verify installed versions against v1.14.1 and v0.30.4`	npm registry listing for Axios; use to confirm current published versions and cross-check against your lock files	HIGH

Framework Mappings

MITRE-ATTACK

- **T1583** — Acquire Infrastructure
- **T1588.001** — Malware
- **T1195.001** — Compromise Software Dependencies and Development Tools
- **T1195.002** — Compromise Software Supply Chain
- **T1136** — Create Account
- **T1078** — Valid Accounts
- **T1486** — Data Encrypted for Impact
- **T1190** — Exploit Public-Facing Application
- **T1213** — Data from Information Repositories
- **T1567.001** — Exfiltration to Code Repository
- **T1657** — Financial Theft
- **T1059** — Command and Scripting Interpreter
- **T1566** — Phishing
- **T1110.003** — Password Spraying

NIST-800-53R5

- **CM-7** — Least Functionality
- **SA-9** — External System Services
- **SR-3** — Supply Chain Controls and Processes
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CP-9** — System Backup
- **CP-10** — System Recovery and Reconstitution
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **AT-2** — Literacy Training and Awareness
- **CA-7** — Continuous Monitoring
- **SI-8** — Spam Protection
- **CM-3** — Configuration Change Control
- **IA-8** — Identification and Authentication (Non-Organizational Users)
- **IR-4** — Incident Handling
- **SR-2** — Supply Chain Risk Management Plan

OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures
- **A07:2021** — Identification and Authentication Failures

CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **16.10** — Apply Secure Design Principles in Application Architectures
- **6.3** — Require MFA for Externally-Exposed Applications
- **6.4** — Require MFA for Remote Network Access
- **6.5** — Require MFA for Administrative Access
- **15.1** — Establish and Maintain an Inventory of Service Providers

ISO-27001-2022

- **A.8.28** — Secure coding
- **A.5.29** — Information security during disruption

- **A.5.21** — Managing information security in the ICT supply chain

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets
- **CC9.2** — Manages risks associated with vendors and business partners

HIPAA-SECURITY

- **164.312(d)** — Person or Entity Authentication
- **164.308(a)(7)(ii)(A)** — Data Backup Plan

NIST-CSF-2

- **RS.MI-01** — Incidents are contained
- **GV.SC-01** — Cybersecurity supply chain risk management program

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1583	Acquire Infrastructure	Resource-Development
T1588.001	Malware	Resource-Development
T1195.001	Compromise Software Dependencies and Development Tools	Initial-Access
T1195.002	Compromise Software Supply Chain	Initial-Access
T1136	Create Account	Persistence
T1078	Valid Accounts	Defense-Evasion
T1486	Data Encrypted for Impact	Impact
T1190	Exploit Public-Facing Application	Initial-Access
T1213	Data from Information Repositories	Collection
T1567.001	Exfiltration to Code Repository	Exfiltration
T1657	Financial Theft	Impact
T1059	Command and Scripting Interpreter	Execution
T1566	Phishing	Initial-Access
T1110.003	Password Spraying	Credential-Access

Sources

Source	URL	Tier
Blog	https://www.crowdstrike.com/en-us/blog/crowdstrike-2026-technology-...	T3
axios Compromised on npm - Malicious Versions Drop Remote ...	https://www.stepsecurity.io/blog/axios-compromised-on-npm-malicious...	T3
Axios NPM Package Compromised: Supply Chain Attack Hits ...	https://www.trendmicro.com/en_us/research/26/c/axios-npm-package-co...	T3
axios npm Compromised: RAT in v1.14.1 & v0.30.4 (2026)	https://phoenix.security/axios-supply-chain-compromise-npm-rat-2026/	T3
Axios Supply Chain Attack: Analysis & Fix - Orca Security	https://orca.security/resources/blog/axios-npm-supply-chain-attack-...	T3

Generated 2026-06-14 13:36 UTC by TJS Security Command Center