NCA-42.091225 – NCERT Advisory – Supply Chain Compromise in npm Ecosystem

Introduction

A critical supply chain compromise has been disclosed in the npm JavaScript ecosystem, involving the compromise of maintainer credentials and the subsequent publication of malicious versions of widely used packages. The incident occurred on September 8, 2025, when attackers gained access to the account of maintainer Josh Junon (qix).

Malicious versions of at least 18 JavaScript packages, including debug, chalk, ansi-styles, and strip-ansi, were uploaded and automatically fetched by developers and CI/CD pipelines. The injected packages contained a browser-based cryptostealer payload capable of silently intercepting and redirecting cryptocurrency transactions, exposing credentials and API keys.

Exploitation required no user interaction beyond package installation, making this a low-complexity and high-impact attack.

Given the widespread usage of these npm packages in enterprise applications, this compromise poses a severe risk of unauthorized code execution, cryptocurrency theft, credential exposure, and downstream system compromise.

Impact

Successful exploitation can lead to:

1. **Cryptocurrency Theft** – Interception and redirection of wallet transactions.
2. **Unauthorized Code Execution** – Execution of injected cryptostealer payloads.
3. **Credential Exposure** – Theft of user credentials, tokens, and API keys.
4. **System Compromise** – Infiltration of applications consuming compromised packages.
5. **Supply Chain Risk** – Propagation of malicious dependencies into downstream systems.

Threat Details

Vulnerability Overview

- **Incident ID**: Supply Chain Compromise – npm ecosystem
- **Affected Component**: npm JavaScript packages (see below)
- **Description**: Attackers uploaded malicious versions of popular npm packages after compromising a maintainer’s account, enabling silent theft of cryptocurrency transactions and credentials.

Attack Complexity & Vector

- **Attack Vector**: Remote (via npm package installation)
- **Attack Complexity**: Low
- **Privileges Required**: None (beyond use of npm ecosystem)
- **User Interaction**: None (after package installation)
- **Estimated CVSS v3.1 Score**: 9.8 (CRITICAL)
- **CWE**: CWE-829 (Inclusion of Functionality from Untrusted Control Sphere), CWE-502 (Deserialization of Untrusted Data)

Indicators of Compromise (IoCs)

Category	Indicator / Value	Description / Notes
Malicious Domain	npmjs[.]help	Typosquatted domain used in phishing campaign to steal maintainer MFA credentials.
Compromised npm Packages	debug@4.4.2, chalk@5.6.1, ansi-styles@6.6.2, strip-ansi@7.1.1, supports-color@10.2.1, ansi-regex@6.2.1, wrap-ansi@9.0.1, color-convert@3.1.1, color-name@2.0.1, color-string@2.1.1, has-ansi@6.0.1, supports-hyperlinks@4.1.1, simple-swizzle@0.2.3, backslash@0.2.1, slice-ansi@7.1.1, chalk-template@1.1.1, is-arrayish@0.3.3, error-ex@1.3.3	Malicious versions injected with cryptostealer payload (published Sept 8, 2025).
Package Metadata	Abnormal publish timestamps: **Sept 8, 2025 (UTC)**	Sudden release of multiple popular packages in short timeframe.
Behavioral IoC	Outbound connections to unknown crypto wallet addresses (via injected browser-based payload)	Indicates theft and redirection of cryptocurrency transactions.
Behavioral IoC	Unexpected harvesting of credentials/API keys from application logs	Suggests malicious package execution.

Exploit Conditions

Exploitation requires:

- Consumption of compromised npm packages during the attack window.
- Automatic fetching of malicious versions via CI/CD pipelines or dependency updates.
- No further user interaction required post-installation.

Recommendations & Mitigation Actions

1. Patch Immediately (Recommended Fix)

- Immediately update to the latest **fixed versions** of all compromised npm packages.
- Run `npm audit` and `npm audit fix` to identify and remediate vulnerable dependencies.

- Pin dependencies to known safe versions in package-lock.json or yarn.lock to prevent accidental reintroduction.
- Rebuild and redeploy all applications that included compromised packages.
- Rotate exposed credentials, tokens, and API keys.

2. Temporary Mitigation (If Patching Not Possible)

- Lock dependencies to verified safe versions and suspend automated updates.
- Continuously monitor build pipelines, logs, and dependency manifests for malicious versions.
- Enforce integrity checks (e.g., package signing, checksums).
- Deploy **Web Application Firewall (WAF)** or runtime monitoring rules to block suspicious browser-based crypto or API manipulation attempts.

3. Strengthen Supply Chain Security Posture

- Enforce multi-factor authentication (MFA) for all npm maintainer accounts.
- Restrict unverified dependency updates.
- Implement continuous monitoring and anomaly detection across build environments.

Monitoring & Detection

- Track npm package installations for unverified versions.
- Monitor application logs for suspicious credential/API calls.
- Use IDS/IPS and SIEM solutions to detect abnormal traffic patterns linked to cryptostealer activity.
- Alert on unexpected outbound cryptocurrency-related connections.

Incident Response & Readiness

- Update incident response playbooks to include supply chain compromise scenarios.
- Validate backups of critical applications and configurations.
- Ensure security teams are briefed on **indicators of compromise (IoCs)** tied to these npm packages.

Patching Summary

Version Category	npm Packages / Components	Status
Vulnerable	Versions listed under "Affected Systems"	Must patch immediately
Secure	Updated to latest fixed versions	No action required

References

- **Supply Chain Attack Targets npm: 2 Billion Weekly Downloads Exposed:** https://securityaffairs.com/182030/security/supply-chain-attack-targets-npm-2-billion-weekly-npm-downloads-exposed.html
- **NVD – CWE-829: Inclusion of Functionality from Untrusted Control Sphere**: https://cwe.mitre.org/data/definitions/829.html
- **NVD – CWE-502: Deserialization of Untrusted Data:** https://cwe.mitre.org/data/definitions/502.html
- **GitHub – debug Issue #1005: Compromised Version Report.** https://github.com/debug-js/debug/issues/1005

Call to Action

The National CERT advises all organizations to:

- Immediately upgrade to fixed versions of all affected npm packages.
- Rebuild and redeploy impacted applications without delay.
- Rotate credentials, API keys, and tokens consumed during the attack window.
- Enhance monitoring of build and deployment environments to prevent recurrence.

Timely remediation is critical to prevent full compromise of applications and systems reliant on npm.