AIRNITE

ZERO DAY

ISSUE 04/2026 · IT SECURITY

Shai-Hulud: the worm that ate its way through JavaScript libraries

In 2025, a self-replicating worm infected hundreds of npm packages, stole developers' credentials and published them on GitHub. How the attack worked and what every development team should do now.

Published: 21 November 2026 Reading time: 6 min Author: Airnite editorial team

SOFTWARE. SECURITY. CODE.

// THE SITUATION IN NUMBERS

500+
compromised packages (wave 1)
CISA, 23 Sep 2025; GitHub removed 500+ packages

≈ 500
packages in wave 2
NCSC New Zealand, 25 Nov 2025

16 Sep 2025
cut-off date for safe versions
CISA: pin to versions released before this date

7 days
lifetime of new npm tokens
measure planned by GitHub (22 Sep 2025)

npm is the central package registry for JavaScript. Many modern web applications pull in numerous third-party building blocks from it. In September 2025, GitHub, which operates npm, reported an attack that spread like a worm from package to package. Its name comes from the novel "Dune": **Shai-Hulud**, the sandworm.

// What happened?

According to GitHub, the company was notified of the attack on 14 September 2025. The self-replicating worm had injected malicious scripts into popular packages. GitHub says it removed **more than 500 compromised packages** from the registry and blocked uploads containing the malware's known indicators. On 23 September 2025, the US agency CISA also issued a warning: the worm had compromised over 500 packages.

A second wave followed in November. On 25 November 2025, New Zealand's NCSC warned of an evolved version affecting almost 500 more packages – with a dangerous new twist: if the malicious code found no usable credentials, it would, according to the NCSC, **wipe all files in the user's home directory**.

// How does the attack work?

The worm exploits trust in package updates. When a developer or a build system installs an infected package, a script runs during installation. According to CISA, it searches the environment for credentials, in particular **GitHub access tokens** and keys for the AWS, Google and Microsoft Azure clouds. It sends the loot to the attackers and also uploads it to a public GitHub repository named "Shai-Hulud".

Then comes the worm part: using the stolen access, the malicious code logs in to npm as the affected developer, according to CISA, injects itself into that developer's other packages and publishes new, infected versions. Anyone who installs them becomes the next victim. In the second wave, according to the NCSC, the malicious code also installed the Bun JavaScript runtime to execute its code and stored the stolen data in randomly named repositories.

// Who is affected?

Every company that develops or builds JavaScript software is affected – which means almost every company with its own web application, online shop or app. Build servers and developer machines holding credentials for the cloud and production are particularly at risk. Installing an infected package there can, in the worst case, hand over the keys to the entire infrastructure.

Companies that only buy software are indirectly affected too: their suppliers work with the same packages. The question for every software partner is therefore: how do you protect your build environment and your credentials?

// What should you do now?

CISA recommends reviewing all projects with npm dependencies, pinning dependencies to safe versions released **before 16 September 2025**, immediately rotating all developer credentials and requiring **phishing-resistant multi-factor authentication** for all developer accounts. The NCSC additionally advises disabling install scripts in CI environments where possible and temporarily freezing package updates.

GitHub announced that it would fundamentally secure package publishing: only with two-factor authentication, with tokens valid for at most **seven days** or via "trusted publishing" directly from verified build systems. The checklist below sums up the steps for your team.

// How does Airnite help?

Airnite develops **software** in a secured build environment: pinned package versions, separate keys for build and production, automatic checks of new dependencies. In a **penetration test**, we check which secrets a compromised package would find in your environment. In **training**, development teams learn to recognise supply chain risks and handle tokens securely.

> Note: research status 21 November 2026. We have not checked whether GitHub has since fully implemented all announced measures.

Checklist: what to do now

[+] Review all projects with npm dependencies, including nested dependencies in package-lock.json or yarn.lock.

[+] Pin dependencies to known, reviewed versions and approve updates deliberately instead of accepting them automatically.

[+] After any suspicion, immediately rotate all developer, GitHub, npm, cloud and CI/CD credentials.

[+] Require phishing-resistant multi-factor authentication (e.g. FIDO keys) for all developer accounts.

[+] Disable package install scripts in CI systems where possible, and run builds without access to production keys.

[+] Enable secret scanning, branch protection and Dependabot security updates on GitHub; remove unnecessary apps and webhooks.

How Airnite helps

- Software & Websites – Websites and web applications: fast, secure, multilingual.
- Penetration Testing & Security – Controlled attacks with a clear report and action plan.
- Training – Your team learns our tools and processes.

Are your developer credentials protected?

Airnite checks your build and development environment: which tokens and cloud keys sit on machines and in CI systems, how packages are pulled in and whether a compromised package could reach your secrets. You get concrete measures – from token rotation to pinned package versions.

Request an initial call · Send an email · Via WhatsApp

Contact

Questions about this article? Message us on WhatsApp: +49 174 6834842

Read online: https://airnite.de/en/magazine/shai-hulud-npm-worm/

Sources

Every fact in this article is based on the sources below. Primary sources are marked.

1. **CISA · Widespread Supply Chain Compromise Impacting npm Ecosystem** · 23 Sep 2025 · PRIMARY SOURCE
   US authority alert: over 500 packages, worm behaviour, recommendations
   https://www.cisa.gov/news-events/alerts/2025/09/23/widespread-supply-chain-compromise-impacting-npm-ecosystem
2. **NCSC New Zealand · New wave of widespread supply chain compromise impacting npm ecosystem** · 25 Nov 2025 · PRIMARY SOURCE
   Authority alert: second wave with almost 500 packages, Bun runtime, wiping of the home directory
   https://www.ncsc.govt.nz/alerts/supply-chain-compromise-impacting-npm-ecosystem/
3. **GitHub Blog · Our plan for a more secure npm supply chain** · 22 Sep 2025 · PRIMARY SOURCE
   Platform operator: removal of 500+ packages, planned measures (2FA, short-lived tokens, trusted publishing)
   https://github.blog/security/supply-chain-security/our-plan-for-a-more-secure-npm-supply-chain/

General editorial information, not legal advice and not an assessment of your individual systems. Status: date of publication.

Airnite GmbH · Am Waldbach 7 · 77654 Offenburg · airnite.de