TECH HIVE ADVISORY

When AI, Trust, and Open Source Collide: Inside the Nx Supply Chain Breach

Imagine this: you update a tool you have used countless times before. No warnings, no glitches. Everything appears normal. But this time, the supply chain update does more than install, it steals. In a matter of minutes, your ***GitHub tokens, SSH keys, and even cryptocurrency wallets*** are quietly siphoned away into repositories you never created. By the next morning, your private code is no longer private.

This nightmare unfolded in August 2025, when attackers compromised the Nx build system, a tool trusted by thousands of developers across the world. What seemed like a routine package update quickly spiraled into one of the most damaging and sophisticated supply chain attacks of the year.

The attack was not simply about malicious code running wild. It revealed just how fragile the foundation of digital trust can be. Artificial intelligence tools, meant to assist developers, were subtly manipulated into aiding the attackers. GitHub's own automated workflows, designed for efficiency and collaboration, were turned against the very people who depended on them. Entire private repositories, many containing sensitive and proprietary work, were dragged into the open without warning.

Here is how trust was broken, and why this matters for every team writing code today.

Supply Chain Breach: The Silent Harvest

The attack unfolded on August 26, 2025, when malicious versions of the Nx package were quietly pushed to npm. Hidden inside was telemetry.js, a tiny script with devastating intent.

Once installed, it went hunting on Linux and macOS systems, searching for cryptocurrency wallets, SSH keys, keystores, environment files, and tokens for GitHub and npm. To add chaos, it even planted a self-destruct command into shell configs, so the next time a victim opened a terminal, their machine could crash instantly.

But the theft did not stop there. The stolen data was wrapped in layers of triple-base64 encoding and then uploaded into the victims' own GitHub accounts. Ghostly repositories with names like s1ngularity-repository, s1ngularity-repository-0, and s1ngularity-repository-1 began appearing by the thousands, each one holding vaults of stolen secrets.

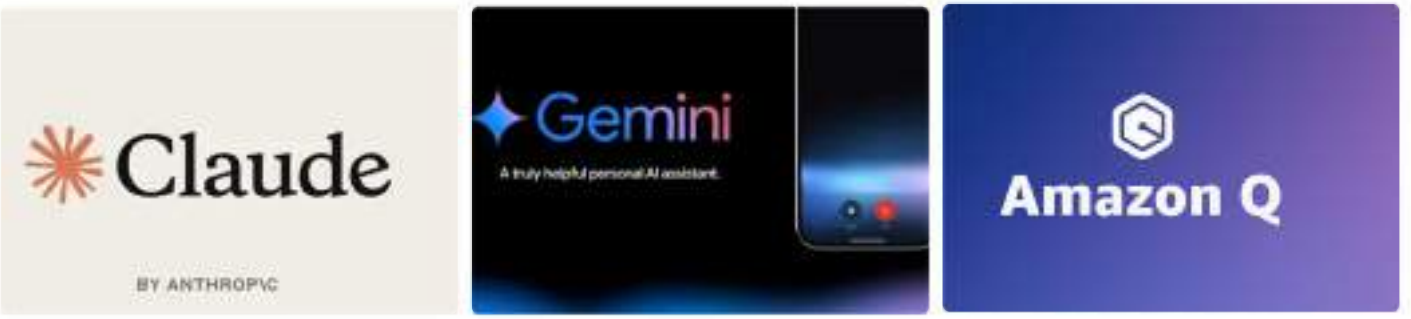
The AI Twist in the Supply Chain: When Helpers Became Traitors

Here is where the attack broke new ground. The malware did not act alone. It hijacked AI command-line tools such as ***Claude, Gemini, and Q.***

By forcing them to run with reckless flags such as ***—yolo, —trust-all-tools, and —dangerously-skip-permissions,*** the attackers turned developer productivity tools into reconnaissance agents.

AI, meant to assist developers, became an unwitting accomplice in one of the year's biggest breaches. Some AI guardrails blocked the attempts, but enough did not. Hundreds of machines were compromised this way.

The very tools meant to protect innovation were manipulated into betraying it.



Phase Two: Private to Public

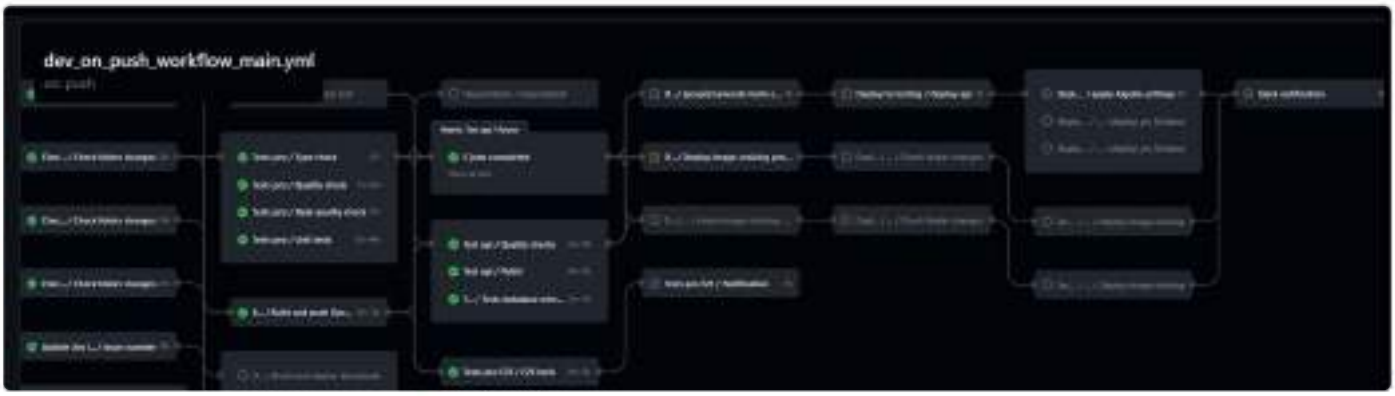
By August 27, 9:00 AM UTC, GitHub stepped in and shut down the attacker-created repositories. But in the eight-hour window before that, the damage had already spread. Stolen data was downloaded and copied.

Then came phase two. Using compromised GitHub tokens, the attackers flipped private repositories to public. Between August 28, 4:00 PM and August 29, 2:00 AM UTC, ***more than 400 organisations and users saw over 5,500 private repositories suddenly exposed.***

This wasn't just theft. Sensitive code was broadcast to the public, entire projects were renamed under the chilling pattern s1ngularity-repository-#5letters#, and intellectual property was left wide open for anyone watching.

For some teams, it meant years of work and competitive advantage spilled into the public domain overnight.

The Supply Chain Backdoor: A Workflow Gone Wrong



At first, poisoned npm packages seemed to explain everything. But on August 28, 9AM UTC, researchers uncovered a deeper flaw: a GitHub Actions workflow misconfiguration.

- It relied on pull_request_target, which runs with elevated permissions.
- PR titles weren't sanitised, allowing malicious code injection.
- Even after removal from the master branch, the workflow lingered in old branches, leaving a hidden backdoor

This meant it wasn't just packages that were compromised. the very supply chain infrastructure meant to protect developers had been turned into a weapon.

The Fallout

By the time the dust settled:

400+ organisations and users hit

5,500+ repositories exposed

Around **20,000** sensitive files lost

Dozens of cloud and npm tokens compromised

1,000+ valid GitHub tokens leaked

And remember, base64 encoding does not hide secrets. It is as private as invisible ink. Anyone who grabbed the repos could decode and abuse the data.

Even after GitHub took action, new malicious repos surfaced. The monitoring continues because exposure lingers.

TECH HIVE ADVISORY

What Security Teams Must Do

- **Remediate:** Purge malicious Nx versions, upgrade to safe releases, delete dropped files, and clean up shell configs.
- **Audit & Detect:** Hunt for rogue repos (s1ngularity-repository*), review GitHub audit logs, and monitor pipelines for suspicious activity.
- **Rotate Credentials:** Revoke and regenerate all tokens (GitHub, npm, SSH, cloud keys). If wallets were exposed, move funds immediately — they can't be rotated.

Lessons From Nx: The Fragility of Supply Chain Trust

This breach was bigger than a poisoned package, it exposed the fragility of trust in the developer ecosystem.

- Developers trusted npm and got poisoned.
- Teams trusted GitHub workflows and got hijacked.
- Users trusted AI tools and saw them weaponised.

The lesson is stark: supply chain security is not just dependency scanning. It's about scrutinising every workflow, every tool, every layer of trust we take for granted.

When trust breaks, as Nx showed, the fallout isn't just vulnerabilities patched. It's secrets leaked, reputations shattered, and futures exposed.

Reference: https://www.wiz.io/blog/s1ngularity-supply-chain-attack

Curated by Oluwafemi Adeleye