ET CISO.in
From The Economic Times

OT Security • 9 Min Read

The simple brilliance of the Axios attack: How a three-hour operation shook the internet

A North Korean threat group stole an npm token from a maintainer through a fake Team link and malicious software, then published a compromised Axios release that stayed live under three hours but may have reached hundreds of thousands of systems. The inciden highlights the risks of single-maintainer dependencies, legacy token-based publishing, a silent transitive exposure across software ecosystems.

Govind Rammurthy • • ETCISO
Published On Apr 15, 2026 at 09:12 AM IST

If you have used a website, a mobile app, or a cloud service in the past decade, there is a very high probability that a software called Axios was

working in the background. Axios is a JavaScript library -- a reusable block of code that developers include in their applications to handle one of the most fundamental tasks in modern software: asking one computer to send information to another over the internet.

The numbers that describe Axios's reach are almost difficult to comprehend. The library is downloaded from its distribution channel -- the npm registry -- over 100 million times every week. It is a direct dependency of more than 174,000 other software packages, meaning that when those packages are used, Axios is used automatically, whether the developer building the final application knows it or not. Security firm Wiz estimates that Axios is present in approximately 80 percent of cloud computing environments worldwide is overwhelming.

The Scouts

The attack that unfolded on 31 March 2026 did not begin that day. By the time anything appeared to go wrong, the work had been underway for at least two months. To understand how it began, it helps to understand how software projects are organised in the open-source world -- and how

much of that organisation is visible to anyone on the internet, including adversaries.

GitHub is the platform where the source code for Axios lives. It is a public repository, meaning that any person anywhere in the world can visit it and read every file, every workflow configuration, every contributor record, and every publish history without creating an account or logging in.

The threat actor behind this attack is tracked by Google's security researchers as UNC1069 and by Microsoft as Sapphire Sleet. They are a North Korean state-sponsored group, active since at least 2018, operating under the revenue-generation mandate of the DPRK government.

When UNC1069's analysts turned their attention to high-value open-source packages, they would have been looking at repositories with large star counts, large download volumes, and -- most importantly -- characteristics in the publish workflow that indicated a stealable credential sitting on a developer's machine. Axios met all three criteria visibly and unambiguously.

The Door That Wasn't There

Saayman was sent a “Join the Teams Meeting link”. When Saayman clicked on the link, he did not join a legitimate Microsoft Teams session. The link he had been sent directed his browser to a fake webpage -- a replica of the Teams meeting interface, hosted on an attacker-controlled domain designed to look like a genuine Microsoft URL at a casual glance. This is a documented UNC1069 technique: constructing lookalike domains that visually pass as legitimate Microsoft infrastructure, combined with the ability to display a malicious web page as a meeting join screen.

Within seconds of landing on this page, a prompt appeared stating that something on his system was “*out of date*” and needed to be updated before the meeting could proceed. This is a technique known in the security industry as ClickFix -- and it is, as of 2025, the single most prevalent initial access method documented by Microsoft, responsible for 47 percent of all attacks tracked by Microsoft Defender Experts, surpassing traditional phishing.

The Scout Inside the Vault

The software that Saayman installed is tracked by Google as WAVESHAPER -- a Remote Access Trojan, or RAT. A RAT is exactly what the name implies: a program that gives its operator remote control over an infected machine, silently and persistently, without the user's knowledge. WAVESHAPER is one of UNC1069's signature tools, with documented evolution dating back to 2023 through multiple North Korean operations.

With the RAT active on Saayman's machine, the attackers now had full access. What they were specifically looking for was the npm authentication token -- the long-lived credential stored in the. npmrc configuration file that would allow them to publish packages to the npm registry on Saayman's behalf.They found it. But they did not use it immediately.

The Staging

Automated security scanners watch the npm registry for newly published packages from unknown accounts. If a brand-new package is injected as a dependency into a major library, it would trigger immediate alerts. The attackers had studied this detection mechanism and planned around it.

At 23:59 UTC on 30 March, a new version -- plain-crypto-js 4.2.1 -- was published. This one was the weapon. It contained an obfuscated JavaScript dropper that, when executed, would silently download and install a platform-specific Remote Access Trojan on whatever machine happened to run it. The package was designed to target Windows, macOS, and Linux simultaneously, with separate payloads for each.

Then, at 00:21 UTC on 31 March 2026, using the stolen npm token from Saayman's machine, the attackers published axios version 1.14.1 -- tagged as the latest release.

This single change to a single configuration file was the entire attack. Axios's own source code was untouched. There was no modified logic, no altered functionality, no visible indication that anything was wrong. The only change was that the package now depended on another package that would execute a malicious script during installation. The attack window was 2 hours and 54 minutes.

Silent Infection at Scale

When a developer runs npm install to use Axios, the package manager automatically downloads not just Axios but every package that Axios depends upon.

During those 2 hours and 54 minutes, every developer who ran npm install in a project with a floating Axios version reference received plain-crypto-js 4.2.1 as part of their installation. The post install hook -- a script that npm runs automatically after installing a package -- then executed the dropper malware. The dropper downloaded the appropriate RAT for their operating system, ran it, then deleted itself and replaced its own configuration file with a clean copy. By the time the installation completed, there was no trace of the attack in the installed files.

The scale of the exposure

Huntress security detected the first infection 89 seconds after

publication. An estimated 600,000 downloads occurred during the exposure window. Huntress confirmed at least 135 compromised systems among its own customer base within hours. Because Axios is a transitive dependency of 174,000 other packages, the exposure extended far beyond direct Axios users -- any CI/CD pipeline, any cloud build environment, any developer workstation that resolved a dependency chain including Axios during that window was potentially affected.

The attack was detected by company whose automated scanner flagged the anomaly within six minutes of publication. The malicious versions were removed from npm approximately three hours after they first appeared. By that point, the damage had been done on hundreds of thousands of machines.

Why the Security Controls Failed

Axios was not an unguarded project. The 1.x branch had implemented OIDC Trusted Publishing -- a mechanism that ties npm releases to specific, verified GitHub Actions workflow runs, generating short-lived tokens that expire after a single use and cannot be extracted from the build environment. On the surface, this appeared to represent a meaningful security improvement. In practice, it protected nothing.

Industry observers have noted that a layered endpoint defence would have offered multiple interception opportunities that the victim's environment lacked entirely.

"When you map this attack against a properly instrumented endpoint, the attack chain collapses at step one. The ClickFix technique depends entirely on the victim executing a command they cannot read -- a

clipboard monitor that inspects and warns before execution eliminates that dependency. If the command still runs, behaviour-based execution control stops the script from reaching out. If the download succeeds, DLP blocks the payload from executing. If the payload runs, egress monitoring blocks the exfiltration. North Korea spent two months on an operation that a layered endpoint defence dismantles in four automated steps. That is the case for eScan Enterprise DLP and Endpoint Protection -- not as a theoretical proposition, but as a direct reading of this incident."

Damage, Repercussions, and the Question of Motive

Every machine that installed the malicious Axios versions during the exposure window should be treated as fully compromised. This is not a precautionary statement -- it is a technical assessment – and the number runs into MILLIONS.

The categories of credentials at risk are significant: npm authentication tokens that could be used for further supply chain attacks, GitHub personal access tokens that could compromise source code repositories, AWS and cloud platform credentials that could give access to production infrastructure, SSH keys that could provide access to servers, and any passwords or session tokens stored in the browser.

The broader repercussion is a structural one. This attack succeeded because a globally critical piece of software was maintained by one person, with no institutional security backing, no mandatory credential rotation policy, and a legacy publish workflow that had never been fully decommissioned. Approximately 80 percent of cloud environments depended on a package whose entire security posture rested on the

personal security hygiene of a single individual who had no reason to expect that a nation-state intelligence operation would spend months building a relationship with him for the specific purpose of stealing a configuration file.

The software industry has responded with commitments to migrate to OIDC-based publishing workflows, enforce short-lived tokens, and add anomaly detection at the registry level for provenance mismatches. These are necessary steps. But they address only the mechanism, not the deeper vulnerability this incident exposed. Even a complete migration to OIDC on GitHub's side leaves the npm registry accepting classic token-based publishes -- because npm does not yet offer enforced OIDC-exclusive publishing at the registry level.

What make this more serious is that a significant proportion of the affected developers may not even know they were exposed! Axios is a transitive dependency of over 174,000 other packages -- meaning that a developer who has never knowingly installed Axios, never referenced it in their own configuration, and would not recognise its name may nonetheless have been running it silently as an indirect requirement of something else entirely!

Given the scale of that silent exposure, it would be no surprise at all to see a sustained wave of credential-based intrusions, supply chain attacks using harvested npm tokens, and highly personalised social engineering campaigns emerge over the coming months -- each one funded by intelligence that North Korea's operators collected in under three hours on a Sunday morning in March.

The author is Govind Rammurthy, CEO & Managing Director, eScan