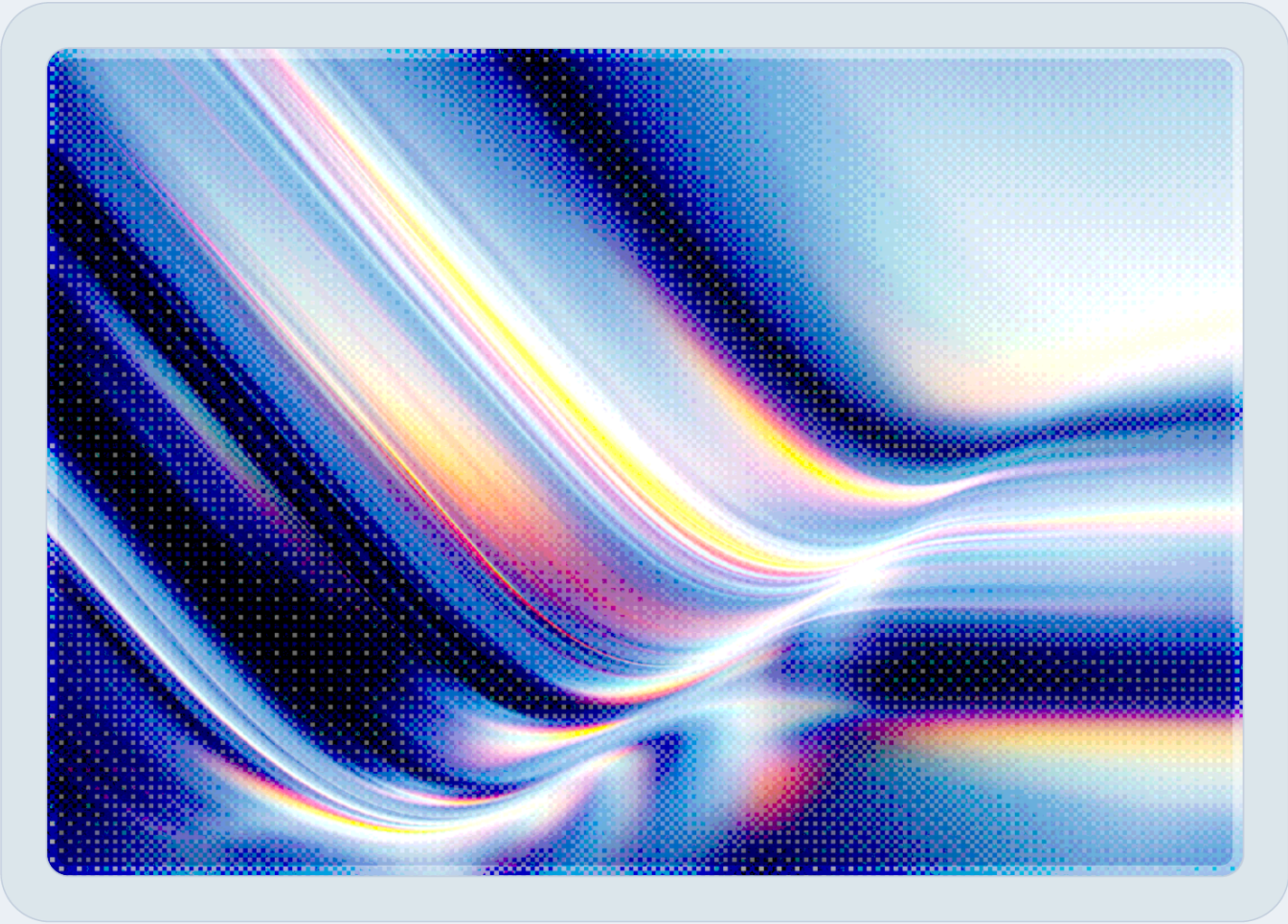
CSAI Foundation | Cloud Security Alliance

CrowdSec Breach: TanStack Fallout Meets Offboarding Failure

2026-09-20

AI-assisted Rapid Research



CSAI

CSA cloud security alliance®

Key Takeaways

CrowdSec, a French cybersecurity vendor known for its crowdsourced threat intelligence network, disclosed on September 17-18, 2026 that attackers had copied roughly 170 of its private GitHub repositories, exposing internal source code, data science models, deployment tooling, and limited personal data on users and early investors [1][2]. The intrusion traces back to the TanStack npm supply chain attack of May 11, 2026 (CVE-2026-45321), in which the threat cluster tracked as TeamPCP published 84 malicious releases across 42 `@tanstack` packages carrying the Mini Shai-Hulud credential-harvesting payload [3][4]. A CrowdSec employee's workstation installed one of the poisoned packages, and the resulting malware exfiltrated a GitHub OAuth token that the attacker used eleven days later to clone the company's private repositories [1][2]. The breach was compounded, and arguably enabled, by an incomplete offboarding process: the employee had departed the company, but CrowdSec left the GitHub token active "so he could finish some work," and the credential was not revoked until three days after the unauthorized cloning had already occurred [1][2]. The incident illustrates a pattern this research series has tracked since the TanStack and Miasma npm campaigns of Q2 2026: a single compromised credential, left alive past the point it was needed, converts a contained upstream supply chain event into a company-specific data breach months later [3].

Background

TanStack maintains widely used open-source JavaScript libraries, including the Router and Start frameworks, that are embedded in a large number of production web applications. On May 11, 2026, TeamPCP compromised the project's GitHub Actions pipeline through a chained exploit involving a `pull_request_target` "Pwn Request" misconfiguration, workflow cache poisoning, and extraction of an OIDC token from GitHub Actions runner memory [3][4]. Within a six-minute window, the attacker used the stolen publishing credentials to push 84 malicious versions across 42 TanStack npm packages, each carrying valid, cryptographically signed provenance attestations that concealed the tampering [3]. The payload, part of the "Mini Shai-Hulud" worm family, searched infected machines for GitHub tokens, npm tokens, SSH keys, and cloud and Kubernetes credentials, then attempted to harvest and exfiltrate them [1][3]. CSA's own research on this period has tied TeamPCP to a broader campaign spanning

March through June 2026 that also compromised the Trivy scanner and the LiteLLM AI gateway, and, in a related but separate incident, Red Hat's `@redhat-cloud-services` npm namespace via the copycat "Miasma" worm [5][8].

One of the organizations whose developers installed a poisoned TanStack package was CrowdSec, a vendor that operates a crowdsourced network for detecting and blocking malicious IP addresses. According to CrowdSec's own incident disclosure, a departing employee's laptop was infected by the malicious package sometime around May 11, and the resulting infostealer captured his GitHub OAuth token [1][2]. The employee's other systems access had reportedly already been removed as part of an offboarding process then underway, but his GitHub token was deliberately kept active because he was still finishing work on the platform [1][2]. On May 22, between 05:52 and 06:01 UTC, an attacker operating from a Toronto-area IP address used that still-valid token to clone approximately 170 of CrowdSec's private repositories [2]. CrowdSec did not revoke the account until May 25, three days after the cloning had already taken place, and the company has stated that the unauthorized access left no trace in the GitHub audit logs it was able to review, meaning it had no visibility into the compromise for nearly four months [1].

The stolen data resurfaced publicly on September 16, 2026, when the archive was posted to a cybercrime forum, prompting a security firm to notify CrowdSec the same day [2]. CrowdSec published a technical postmortem the following days confirming the TanStack connection and detailing the scope of exposure [2].

Security Analysis

The exposed archive reportedly included the source code for CrowdSec's web console, data science scripts and detection models, internal automation and deployment tooling, and the "consensus algorithm" the company uses to determine when an IP address should be added to its shared blocklists [1][2]. More than 130 of the copied repositories were already public, but the private subset contained proprietary detection logic that CrowdSec has not otherwise disclosed [2]. Beyond source code, the leak exposed the email addresses of 83 CrowdSec users, along with the names, email addresses, and investment details of 51 individuals who were prospective investors during a 2020 funding round [1][2]. CrowdSec has stated that no customer data, production infrastructure, or operational systems were accessed, and that forensic review found no evidence the attacker made any commits or altered code, infrastructure, or CI/CD configuration during the intrusion [2]. A separate access attempt on August 17, 2026, in which someone used an exposed AWS credential to call `GetCallerIdentity` and list SNS topics, suggests the attacker retained and periodically tested other credentials harvested in the same campaign well after the initial data theft [2][3].

This incident is instructive less for its technical novelty than for what it reveals about the gap between supply chain compromise and its downstream consequences. The initial TanStack compromise was a widely reported, high-severity event (CVSS 9.6) that prompted broad remediation guidance across the industry within days [3][4]. Yet the CrowdSec breach did not surface until more than four months later, and only because the stolen data was independently posted for sale rather than through CrowdSec's own detection. That lag reflects two compounding weaknesses. First, an OAuth token that is scoped to a departing employee's account but deliberately kept alive past its operational need typically loses its accountable, actively monitored owner: in this case, nobody in fact noticed the unusual activity from an account whose user had already left. Second, CrowdSec's acknowledgment that the intrusion left no visible trace in the GitHub logs it could review points to a detection gap around bulk repository cloning by an authenticated OAuth application, an action that does not require a commit, a pull request, or any other change to observable code state. In CSA's assessment, multifactor authentication, which many organizations treat as the primary defense against account takeover, is structurally incapable of stopping this class of attack: the token was already authorized and already stolen from a live session, so no subsequent authentication challenge could have intervened.

The broader pattern connects this incident to the TeamPCP and Miasma campaigns analyzed in CSA's prior npm supply chain research: credential-harvesting worms do not need to compromise a target organization directly. It is sufficient for the worm to compromise a single developer's machine anywhere in the software supply chain and wait for that developer's residual, unrevoked access to become useful [3]. The CrowdSec case shows that this exposure window can extend for months, and that in this instance the trigger for discovery was the stolen material's appearance for sale rather than internal detection — a dynamic CSA's prior research on the TeamPCP and Miasma campaigns suggests may not be unique to this incident [5][8].

Recommendations

Immediate Actions

Organizations that use TanStack packages, or any dependency published during the May 11, 2026 compromise window, should confirm that affected developer workstations have been scanned for the Mini Shai-Hulud payload and that all credentials those machines could access, including GitHub OAuth and personal access tokens, npm tokens, SSH keys, and cloud provider credentials, have been rotated regardless of whether misuse has been observed [3][4]. Security teams should also audit GitHub

organization settings for any OAuth application or personal access token still associated with a departed employee, since CrowdSec's experience shows that such tokens can remain valid, undetected, and unaccountable well past an employee's exit date.

Short-Term Mitigations

Offboarding procedures should be restructured so that access revocation is atomic rather than staged: a departing employee's credentials, tokens, and OAuth grants should be revoked in a single coordinated action rather than left partially active to accommodate transition work, since a partial revocation creates exactly the kind of orphaned, unmonitored credential that enabled this breach. Where legitimate business need requires continued access after departure, that access should be reissued under a time-boxed, separately monitored service credential rather than the departing individual's personal token, and it should carry an explicit expiration date enforced by policy rather than manual follow-up. Organizations should also extend GitHub audit log retention and alerting to flag bulk repository cloning or unusual data-volume access by any single OAuth token, since this activity pattern is a stronger and earlier signal of compromise than the absence of code changes.

Strategic Considerations

At a program level, this incident underscores that identity lifecycle management for machine-usable credentials, OAuth tokens, personal access tokens, and API keys, deserves the same governance rigor that organizations increasingly apply to human account offboarding. Least-privilege and just-in-time access principles from Zero Trust architecture apply directly here: a credential that persists only because revoking it is inconvenient is a standing liability, and its blast radius is not bounded by the departed employee's intent but by whatever else compromises that credential later. Organizations dependent on open-source JavaScript tooling should also treat the possibility of supply chain credential theft as a standing condition rather than an isolated event, given that TeamPCP-linked campaigns compromised multiple ecosystems across March through June 2026 alone, and should build detection capability for delayed, second-order exploitation of credentials stolen months earlier [3].

CSA Resource Alignment

This incident sits at the intersection of three threads in CSA's 2026 research: npm supply chain compromise, identity lifecycle governance, and OAuth token abuse as a standing breach vector. CSA's whitepaper, "npm Supply Chain Under Siege: TeamPCP, Miasma, and npm v12" [5], directly analyzes the TeamPCP campaign and the underlying TanStack CI/CD compromise that produced the credential theft

behind this breach, including the Mini Shai-Hulud payload family that infected the CrowdSec employee's machine. CSA's research note "Miasma: Red Hat npm Supply Chain Worm" [8] documents the copycat campaign against Red Hat's `@redhat-cloud-services` namespace referenced above, and CSA's research note "ShinyHunters' OAuth Pivot: A Year of SaaS Supply-Chain Breaches" [7] traces the same underlying pattern identified in CrowdSec: a durable, overlooked OAuth token becoming the vector for a downstream breach long after the credential was first compromised. One of [5]'s key findings, that cryptographic provenance attestations like SLSA and Sigstore protect against build tampering but not against credential theft from an already-authorized account, closely mirrors the failure mode the CrowdSec breach later demonstrated at the organizational level: the stolen OAuth token was fully "legitimate" from GitHub's perspective, and no provenance or signing control could have flagged its misuse.

The offboarding failure at the center of this incident is fundamentally an identity governance lapse rather than a technical exploit, and CSA's Zero Trust Guiding Principles [6] speaks directly to the underlying control failure. The guidance's emphasis on least-privilege, time-bound access and continuous verification, rather than standing trust granted to an identity because it was once legitimate, describes the practice CrowdSec did not follow when it left a departing employee's token active without a defined expiration or monitoring plan. Organizations seeking to prevent a recurrence of this pattern should treat OAuth token lifecycle management as a Zero Trust identity control, not merely an HR offboarding checklist item, and should reference CSA's AI Controls Matrix (AICM) supply chain and identity and access management domains when assessing whether their own dependency and credential governance would have caught a comparable exposure.

References

[1] The Hacker News. "CrowdSec Says TanStack npm Attack Led to Copy of 170 Private GitHub Repositories." The Hacker News, September 2026.

[2] CrowdSec. "TanStack Supply Chain Attack Analysis." CrowdSec Blog, September 2026.

[3] Cyber Security News. "TanStack Supply Chain Attack Lets Hackers Steal 170 Private CrowdSec GitHub Repositories." Cyber Security News, September 2026.

[4] Broadcom. "CVE-2026-45321 - TanStack npm Supply Chain Compromise." Broadcom Security Center, 2026.

[5] Cloud Security Alliance. "npm Supply Chain Under Siege: TeamPCP, Miasma, and npm v12." CSA Labs, June 2026.

[6] Cloud Security Alliance. "Zero Trust Guiding Principles." Cloud Security Alliance, 2026.

[7] Cloud Security Alliance. "ShinyHunters' OAuth Pivot: A Year of SaaS Supply-Chain Breaches." CSA Labs, July 2026.

[8] Cloud Security Alliance. "Miasma: Red Hat npm Supply Chain Worm." CSA Labs, June 2026.

