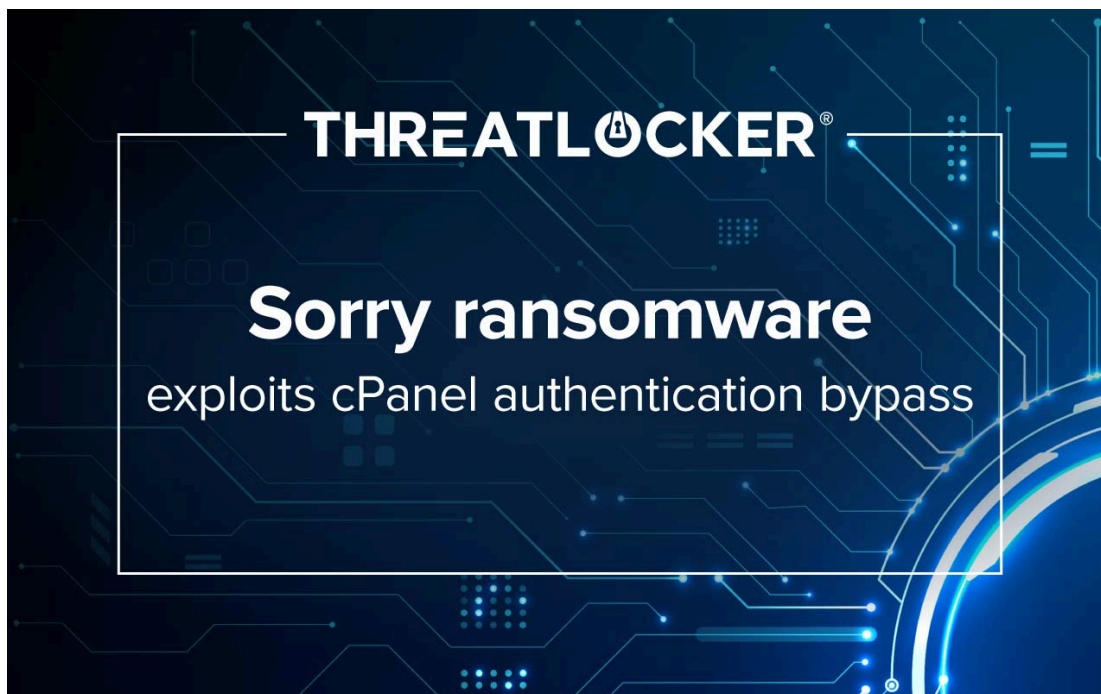


TeamPCP supply chain attack hits TanStack

 threatlocker.com/blog/teampcp-supply-chain-attack-hits-tanstack

May 15, 2026



May 15, 2026

Written by:

ThreatLocker Threat Intelligence

TanStack and more npm packages compromised

On May 11, 2026, attackers published 84 malicious versions across 42 @tanstack/* npm packages. The packages were authenticated through TanStack's legitimate GitHub Actions OIDC trusted-publisher binding, but the publish workflow itself was not directly modified and no npm tokens are known to have been stolen.

Instead, the attacker chained a `pull_request_target` Pwn-Request pattern, GitHub Actions cache poisoning, and runtime extraction of an OIDC token from GitHub Actions runner memory to publish credential-stealing Mini Shai-Hulud payloads.

Mini Shai-Hulud is a cross-ecosystem software supply chain attack campaign attributed to TeamPCP by researchers.

The May 11 TanStack wave primarily affected npm packages, with related propagation into PyPI packages such as `mistralai==2.4.6` and `guardrails-ai==0.10.1`. Broader Mini Shai-Hulud activity has also been reported outside npm, but this TanStack-focused report should distinguish the May 11 npm/PyPI wave from earlier or adjacent registry compromises.

Technical breakdown of Mini Shai-Hulud supply chain attack

This supply chain attack is attributed to TeamPCP, extending their slew of attack chain compromise campaigns. TeamPCP exploited a chain of three vulnerabilities in the CI/CD pipeline: Pwn Request through `pull_request_target`, GitHub Actions cache poisoning, and runner memory OIDC token extraction. This combination resulted in malicious versions being published with valid level 3 SLSA provenance.

Staging: Pwn request

TanStack has attributed this detection to StepSecurity researcher, Ashish Kurmi, in their postmortem.

In his report he states that the payload is staged in commit (79ac49ee), a fork of `Tanstack/router` by the account `voicproducoes` (ID: 269549300). The account was briefly mentioned in TanStack's postmortem report, but the only mentions of the commit were under the IOC fingerprint section and as an open question: "Can we identify any other fork in the `TanStack/router` fork network that contains the orphan payload commit?"

This commit appears to have been orphaned as it can no longer be found, and as such, cannot be inspected.

We were able to confirm that the attacker had created a fork of `TanStack/router` on 2026-05-10 under the account `zblgg`. The fork was named `zblgg/configuration` to avoid being spotted by `fork-list` searches. This fork had the prefix `[skip ci]` added to suppress automated CI on push by an impersonating Anthropic identity `claude` `<claude@users.noreply.github.com>` on commit 65bf499d.

The threat actor account `zblgg` opened PR #7378 against `TanStack/router#main` which used the `pull_request_target` trigger to leverage the `bundle-size.yml` workflow. This trigger runs the malicious fork commit with the base repo's security context.

Injection: Cache poisoning

TanStack's `actions/cache@v5` can write cache entries during its post-job save step, since cache writes use a runner-internal token rather than the workflow's `GITHUB_TOKEN`, meaning cache mutations are not prevented by `permissions: contents: read`.

Secondly, GitHub Actions caches are shared at the repository level, including between `pull_request_target` runs and trusted workflows on `main`, so an untrusted PR can poison cache entries that will later be consumed by production workflows.

In this case, a malicious `vite_setup.mjs` intentionally modified the `pnpm-store` under the exact cache key later used by the legitimate `release.yml` workflow. When the PR job was completed, the poisoned cache was saved, and the next `main` release run restored it as expected.

Publishing: Memory extraction

The malicious packages carried valid SLSA Build Level 3 provenance attestations, meaning they appeared to originate from the expected GitHub Actions build pathway.

They were able to achieve this by publishing these malicious packages through POST requests sent directly to `registry.npmjs.org` with stolen OIDC tokens, bypassing the typical "Publish Packages" step of the workflow and satisfying the required security measures. These OIDC tokens exist in the process memory of `Runner.Worker` and are relatively easy to capture by reading `/proc/<pid>/mem`.

Shai Hulud credential-stealing worm

The impact and propagation mechanism of this attack chain is the Mini Shai-Hulud credential-stealing worm.

Earlier Shai-Hulud waves were observed in September and November 2025, with Mini Shai-Hulud activity continuing in April 2026 and expanding again in

the May 11 TanStack wave. The worm is notable for using stolen or hijacked publishing authority to publish additional malicious package versions, allowing compromise to spread through trusted package ecosystems.

Because this worm's spread depends on the npm registry, it does not require any single running execution parent to spread and simply waits for users to download it from a compromised package. The ease of use granted through the npm workflow allows the worm to spread quicker than defenders can detect it, and new propagation chains often result in thousands of compromised users (often in a matter of hours) before the attack is detected and mitigated.

Many packages also have others required as dependencies, which exponentially increases the victim count if a package in this chain is breached.

The worm also includes separate workflows for captured GitHub credentials and has become notorious for exfiltration by publishing public repositories containing CI/CD secrets, workflows, and captured credentials. These public packages are quickly noticeable by their descriptions, often matching terms such as "Sha1-Hulud: The Second Coming", "Shai-Hulud Migration", and "Shai-Hulud: Here We Go Again". Several unmaintained repositories currently have sensitive documents and secrets active under these terms.

Mini Shai Hulud shows several advancements from its parent variants and implements preinstall files and import-time hooks to execute an obfuscated JavaScript infostealer. Obtained secrets such as developer tokens, cloud credentials, and CI/CD workflows are exfiltrated by creating public Github repositories following similar descriptions to previous versions.

Mitigation and response to Mini Shai-Hulud

Immediate triage efforts should start with the identification of exposure in lockfiles and installed modules.

Files `package-lock.json`, `pnpm-lock.yaml`, `yarn.lock`, CI build logs, and artifact caches should be searched for any of the affected package names and their affected versions. In addition, files named `router_init.js` or `tanstack_runner.js` should be purged from repositories and package artifacts.

Any affected hosts, including development hosts, and CI runners that installed a compromised version should be treated as fully compromised, especially if package scripts executed successfully, and any secrets from CI, cloud, Vault, Kubernetes, or package registries, SSH keys, and local developer files should be treated as compromised.

On Linux hosts, persistence scripts may be deployed under directories `.vscode/` and `.claude/`, or as Linux user services. On macOS hosts, LaunchAgents to maintain persistence may be present under the name `gh-token-monitor`. These persistence mechanisms should be removed before token revocation because the `gh-token-monitor` dead man's switch reportedly attempts to run `rm -rf ~/`, deleting the affected user's home directory if token revocation is detected.

To harden CI/CD environments, it is recommended to disable lifecycle scripts, enforce lockfile-only installs, add release-age cooldowns for newly published package versions, scope npm trusted publishing to a specific protected branch and workflow file, pin GitHub Actions to full commit SHAs, and to clear action caches after suspected compromise.

IOCs

File hashes - SHA-256

`router_init.js`:

`ab4fcadaec49c03278063dd269ea5eef82d24f2124a8e15d7b90f2fa8601266c`

`tanstack_runner.js`:

`2ec78d556d696e208927cc503d48e4b5eb56b31abc2870c2ed2e98d6be27fc96`

`@tanstack/setup package.json`:

`7c12d8614c624c70d6dd6fc2ee289332474abaa38f70ebe2cdef064923ca3a9b`

`setup.mjs`:

`2258284d65f63829bd67eaba01ef6f1ada2f593f9bbe41678b2df360bd90d3df`

Malicious package / dependency indicators

`"@tanstack/setup"`:

`"github:tanstack/router#79ac49eedf774dd4b0cfa308722bc463cfe5885c"`

Malicious orphan commit: 79ac49eedf774dd4b0cfa308722bc463cfe5885c

Payload file: router_init.js

Runtime artifact: router_runtime.js

Payload runner: tanstack_runner.js

PyPI payload staging path: /tmp/transformers.pyz

PyPI affected package: mistralai==2.4.6

PyPI affected package: guardrails-ai==0.10.1

Network indicators

filev2[.]getsession[.]org

seed1[.]getsession[.]org

seed2[.]getsession[.]org

seed3[.]getsession[.]org

api[.]masscan[.]cloud

git-tanstack[.]com

git-tanstack[.]com/tmp/transformers.pyz

git-tanstack[.]com/transformers.pyz

github[.]com/oven-sh/bun/releases/download/bun-v1.3.13/

83.142.209.194

169.254.169.254

169.254.170.2

127.0.0.1:8200

Cloud / Metadata / Secret Access Targets

169.254.169.254 - AWS EC2 IMDS queried for IAM role credentials

169.254.170.2 - ECS/Fargate task metadata credentials

127.0.0.1:8200 - Local HashiCorp Vault access

TLS / certificate indicator

TLS certificate pin: CN=seed1.getsession.org, O=Oxen Privacy Tech Foundation, expires 2033

Persistence artifacts

.claude/settings.json

.claude/router_runtime.js

.claude/setup.mjs

.vscode/tasks.json

.vscode/setup.mjs

~/Library/LaunchAgents/com.user.gh-token-monitor.plist

~/config/systemd/user/gh-token-monitor.service

~/local/bin/gh-token-monitor.sh

.github/workflows/codeql_analysis.yml

Persistence / execution descriptions

.claude/settings.json - SessionStart hook that re-executes malware on Claude Code session start

.vscode/tasks.json - folderOpen task that re-executes malware when VS Code opens

.claude/router_runtime.js - Bun payload dropped for persistence

.claude/setup.mjs - shared setup script

.vscode/setup.mjs - shared setup script

~/Library/LaunchAgents/com.user.gh-token-monitor.plist - macOS GitHub token monitoring service

~/.config/systemd/user/gh-token-monitor.service - Linux GitHub token monitoring service

~/.local/bin/gh-token-monitor.sh - Linux GitHub token monitoring script

.github/workflows/codeql_analysis.yml - injected workflow used to exfiltrate repository secrets on push or deployment

GitHub / campaign indicators

Spoofed commit author: claud@users.noreply.github.com

Dead-drop commit message: chore: update dependencies

Token threat string:

IfYouRevokeThisTokenItWillWipeTheComputerOfTheOwner

Campaign PBKDF2 salt: svksjrhjkcejg

Worm marker repo description: A Mini Shai-Hulud has Appeared

Reported attacker account: voicproducoes

Reported fork: voicproducoes/router

Also reported fork / infrastructure: zblgg/configuration

Dune-themed branch markers: fremen, sandworm, harkonnen, atreides, melange, tleilaxu

RELATED blogS
