TeamPCP: Trivy Supply Chain Attack and Kubernetes Wiper

How a Compromised Vulnerability Scanner Became a CI/CD Weapon

Unofficial AI-assisted Research

Cloud Security Alliance AI Safety Initiative

2026-03-24

Key Takeaways

- On March 19, 2026, the threat actor cluster TeamPCP force-pushed a malicious `v0.69.4` release tag to the Aqua Security Trivy repository, injecting a three-stage credential stealer into a vulnerability scanner used in over 10,000 CI/CD pipelines worldwide [1][17].
- The attack exploited an incomplete credential rotation following a February 2026 GitHub Actions misconfiguration, demonstrating that partial remediation can leave organizations still exposed to the original vulnerability, as attackers exploit residual access before the gap is fully closed.
- Stolen CI/CD credentials were weaponized within 96 hours to compromise the Checkmarx AST GitHub Action, illustrating how a single supply chain breach cascades to downstream tools [2].
- TeamPCP's follow-on payload, CanisterWorm, used an Internet Computer Protocol (ICP) blockchain canister as a command-and-control resolver — a novel evasion technique that resists conventional infrastructure takedowns, as the canister cannot be deprovisioned via abuse notices or registrar action, though DNS egress blocking of known ICP polling domains remains a useful first-line defense [3].
- Kubernetes clusters on systems matching Iranian locale were subjected to a destructive wiper DaemonSet that deleted all host filesystem contents and forced a node reboot, rendering affected infrastructure unrecoverable [4].
- Any pipeline that ran Trivy versions `0.69.4`-`0.69.6` or referenced `aquasecurity/trivy-action` between March 19–23, 2026 should be treated as fully compromised; all secrets, tokens, and cloud credentials from those runs must be rotated immediately.

Background

The Threat Actor: TeamPCP

TeamPCP is a cloud-native criminal threat cluster whose public Telegram channel — operating under aliases including **DeadCatx3**, **PCPcat**, **PersyPCP**, and **ShellForce** — became active no later than July 2025 [5]. With more than 700 channel members and a track record of publishing stolen data from victims across Canada, Serbia, South Korea, the UAE, and the United States, the group has established itself as a persistent criminal operation rather than a one-time campaign. Security researchers characterize the group's tradecraft as the large-scale automation of well-documented attack techniques, using copied, lightly modified, and AI-assisted code to industrialize known cloud misconfigurations [5]. Their business model combines credential theft, ransomware, extortion, cryptocurrency mining, and proxy infrastructure rental.

TeamPCP demonstrated its cloud-native focus in December 2025, when the group launched a worm-driven campaign targeting exposed Docker APIs, Kubernetes clusters, Ray dashboards, and Redis servers at scale. That campaign exploited CVE-2025-55182 ("React2Shell," CVSS 10.0), a pre-authentication remote code execution vulnerability in React Server Components, and CVE-2025-29927, a React/Next.js middleware bypass. The December 2025 wave compromised more than 60,000 servers, with Azure and AWS environments accounting for approximately 97% of victims [5]. The December 2025 campaign likely established the infrastructure and tooling that TeamPCP subsequently turned against the software supply chain, though direct infrastructure reuse in the March 2026 Trivy attack has not been independently confirmed.

Trivy and the CI/CD Trust Problem

Trivy is an open-source vulnerability scanner maintained by Aqua Security and widely embedded in CI/CD pipelines as a security gate. Its role as a trusted security primitive — something pipelines invoke specifically to protect against compromise — makes it an especially high-value supply chain target. When a vulnerability scanner is backdoored, the attack surface extends to every pipeline that relies on it, and the malicious code runs with the elevated permissions that security tooling routinely requires. At the time of the breach, more than 10,000 GitHub workflow files referenced `aquasecurity/trivy-action` directly [1][17].

The attack's prerequisites were established months before the March 2026 payload deployment. In late February 2026, a threat actor exploited a misconfiguration in Trivy's GitHub Actions environment to extract a privileged access token. The Trivy team disclosed the incident on March 1, 2026, and performed credential rotation — but the rotation was incomplete, leaving residual access through credentials that remained valid. TeamPCP exploited this gap on March 19, 2026, at 17:43:37 UTC [1][6].

Security Analysis

Stage One: Repository Takeover

The initial payload delivery involved a force-push of the malicious `v0.69.4` tag to the Trivy repository, triggering an automated release pipeline. The attack simultaneously compromised three components: the Trivy core scanner binary for Linux x86-64, Windows x86-64, and macOS ARM64; 75 of 76 version tags in `aquasecurity/trivy-action`; and seven tags in `aquasecurity/setup-trivy` [1][6]. Attackers impersonated legitimate Trivy contributors through spoofed commits and weaponized the `aqua-bot` and `Argon-DevOps-Mgt` service accounts. The malicious releases were indistinguishable from legitimate ones in pipeline logs because the credential stealer ran before the normal Trivy scan, and the scan itself completed successfully — pipelines received no signal that anything had gone wrong.

Stage Two: CI/CD Credential Harvesting

The injected payload, identified as the "TeamPCP Cloud Stealer," operated in three stages. In the collection phase, it scraped `Runner.Worker` process memory via `/proc/<pid>/mem`, searching for the structured secret pattern `{"value":"<secret>","isSecret":true}` that GitHub Actions uses internally. This approach bypassed masking and environment variable inspection by targeting secrets in memory rather than the environment. The stealer also harvested SSH keys, cloud credentials for AWS, GCP, and Azure, Kubernetes service account tokens, cryptocurrency wallets from more than 50 file paths, and webhook URLs for Slack and Discord. On AWS-hosted runners, it queried the Instance Metadata Service at `169.254.169.254` to retrieve IAM role credentials [1][6].

In the encryption phase, collected credentials were bundled as `tpcp.tar.gz` and protected with AES-256-CBC and RSA-4096 hybrid encryption. In the exfiltration phase, the bundle was uploaded via `curl POST` to the typosquatted domain `scan.aquasecurtiy[.]org` (note the transposed `i` and `t` in "security"), resolving to `45.148.10.212` (TECHOFF SRV LIMITED, Amsterdam). A Cloudflare Tunnel endpoint served as an alternative exfiltration channel specifically for GitHub-hosted runners. As a tertiary fallback, the malware created a repository named `tpcp-docs` in the victim's GitHub account containing the encrypted secrets archive [1][6].

On machines outside GitHub Actions environments, the binary dropped a Python persistence dropper at `~/.config/systemd/user/sysmon.py` that polled the ICP canister command-and-control infrastructure every five minutes.

Stage Three: Cascading Compromise

The stolen credentials proved immediately actionable. On March 20, 2026, within 24 hours of the initial Trivy breach, CanisterWorm was deployed to the npm ecosystem using stolen npm publish tokens [3]. On March 22, 2026, additional malicious Trivy Docker Hub images were published as versions `0.69.5` and `0.69.6`. In a two-minute automated burst on March 22 at 20:31–20:32 UTC, all 44 repositories in Aqua Security's internal `aquasec-com` GitHub organization were defaced — renamed with the prefix `tpcp-docs-` — including proprietary source for Tracee, internal Trivy forks, CI/CD pipelines, Kubernetes operators, and internal knowledge bases [6]. The attackers had confirmed write access by creating and silently deleting a test branch seven hours earlier.

Approximately four days after the initial Trivy breach, the structurally identical credential stealer appeared in Checkmarx's `ast-github-action` version `2.3.28`, using stolen tokens acquired from victim CI/CD runners to compromise a second major security toolchain [2]. The only differences between the Trivy and Checkmarx payloads were the entry point (`entrypoint.sh` versus `setup.sh`) and the exfiltration domain (`scan.aquasecurtiy[.]org` versus `checkmarx[.]zone`, resolving to `83.142.209.11`). Both used identical packaging (`tpcp.tar.gz`) and curl upload patterns.

CanisterWorm and Blockchain-Based C2

CanisterWorm's use of an immutable on-chain canister as a C2 resolver distinguishes it from conventional domain-fronting and fast-flux DNS techniques by eliminating the hosting provider as a remediation lever. Distributed via malicious `postinstall` hooks in compromised npm packages, the worm installed a Python backdoor that persisted via a systemd user service named `pgmon` and polled an Internet Computer Protocol canister at `tdtqy-oyaaa-aaaae-af2dq-cai.raw.icp0.io` approximately every 50 minutes [3][4]. The ICP canister — a tamperproof smart contract on the Internet Computer blockchain — served as a dead-drop resolver: it stored a URL pointing to the actual payload server, and attackers could update that URL at any time while the C2 channel remained available indefinitely and resistant to conventional takedown requests, as no single hosting provider or registrar controls canister availability. Blocking the ICP canister domain required coordinated action at the blockchain layer, which took days rather than hours [3].

CanisterWorm's propagation mechanism targeted every npm package owned by a compromised publisher. The `deploy.js` component queried the npm registry with stolen tokens, identified all packages associated with the compromised account, incremented the patch version number, and published the malicious update via `npm publish`. A later mutation found in `@teale.io/eslint-config` versions 1.8.11 and 1.8.12 automated this propagation without requiring manual attacker intervention. At least 47 npm packages were compromised in the initial wave — the consolidated figure per Aikido's analysis [3] and later reporting [15], with Socket.dev reporting 29 packages at an earlier pre-propagation snapshot [7] — expanding to 141 malicious package artifacts across 66 unique packages in the namespaces `@emilgroup` and `@teale.io` [3][15].

The Kubernetes Wiper

After establishing persistence via CanisterWorm, TeamPCP deployed a destructive follow-on payload to Kubernetes clusters where collected credentials granted sufficient access. The wiper checked two conditions to determine its Iranian targeting: the presence of `Asia/Tehran` or `Iran` in `/etc/timezone` or `timedatectl` output, and the `LANG` environment variable matching `fa_IR` [4].

On Kubernetes clusters matching Iranian locale indicators, the payload deployed a DaemonSet named `host-provisioner-iran` in the `kube-system` namespace, with tolerations configured to schedule across all nodes including the control plane. Each node ran a privileged Alpine container named `kamikaze` that mounted the host root filesystem to `/mnt/host` and executed a find-based deletion of all top-level host directories followed by a forced reboot. This sequence effectively bricked every node in the cluster by destroying the underlying host OS while the container remained briefly alive to trigger the reboot. On non-Iranian clusters, a functionally equivalent DaemonSet named `host-provisioner-std` was deployed instead, installing CanisterWorm as a systemd service for persistent backdoor access rather than destruction. On Iranian non-Kubernetes Linux hosts, the payload executed `rm -rf / --no-preserve-root` directly [4].

The geopolitically targeted nature of the wiper — delivering destruction to Iranian infrastructure while deploying persistence tooling everywhere else — suggests either a nation-state nexus or a criminal actor with nation-state clients. Whether TeamPCP operates under state direction, accepts state-aligned commissions, or made independent targeting decisions cannot be determined from publicly available information.

Recommendations

Immediate Actions

Any organization whose CI/CD pipelines invoked Trivy, `aquasecurity/trivy-action`, or `aquasecurity/setup-trivy` between March 19 and March 23, 2026 should treat those runs as fully compromised. The following actions are required and should be executed in parallel where possible.

Assume compromise and rotate all exposed credentials. All secrets, tokens, cloud credentials, SSH keys, and API keys accessible during affected pipeline runs must be rotated immediately, regardless of whether anomalous behavior was observed. This includes long-lived AWS access keys, GCP service account keys, Azure client secrets, and any npm publish tokens that ran through compromised environments.

Pin action versions to full commit SHAs. Replace all `uses: aquasecurity/trivy-action@v0.X.X` references with pinned commit SHA references (e.g., `uses: aquasecurity/trivy-action@<sha>`). Version tags in GitHub Actions are mutable and can be force-pushed without notice.

Verify Trivy binary integrity. Confirm that the Trivy binary version in use is `0.69.3` or earlier, or `0.69.7` or later following Aqua Security's confirmed remediation. Validate binary checksums against those published by Aqua Security through out-of-band channels. The hashes for known-malicious Linux, Windows, and macOS binaries are listed in the Indicators of Compromise section.

Audit npm package dependencies. Review all npm dependencies for packages in the `@emilgroup` and `@teale.io` namespaces, and check `postinstall` scripts across the dependency tree for `pgmon` references or ICP canister polling code.

Block or monitor known malicious domains and IPs. Add `scan.aquasecurtiy[.]org` (45.148.10.212), `checkmarx[.]zone` (83.142.209.11), and `tdtqy-oyaaa-aaaae-af2dq-cai.raw.icp0.io` to DNS blocklists and network egress deny lists. Note that ICP canister URLs may change; monitor for outbound connections matching the ICP canister HTTP pattern.

Hunt for persistence artifacts. Search all developer workstations, CI/CD runners, and Kubernetes nodes for the files `/var/lib/pgmon/pgmon.py`, `/etc/systemd/system/pgmonitor.service`, `~/.config/systemd/user/sysmon.py`, `/tmp/pglog`, and `/tmp/.pg_state`. On Kubernetes clusters, audit all DaemonSets in `kube-system` for the names `host-provisioner-iran` and `host-provisioner-std`.

Short-Term Mitigations

Organizations should implement cryptographic verification of all third-party GitHub Actions and pipeline tools as a standard practice, not an exception. Sigstore's Cosign and GitHub's artifact attestation framework provide mechanisms to verify that a binary or container image was produced by the expected build pipeline and was not tampered with in transit. The Trivy attack exploited the absence of this verification: pipelines consumed a malicious binary because they trusted the tag, not the cryptographic chain of custody.

CI/CD runners should operate under least-privilege principles with secrets scoped narrowly to individual workflow steps. The TeamPCP Cloud Stealer succeeded in harvesting broad credentials because GitHub Actions runners routinely hold all pipeline secrets in memory simultaneously. Vault-based secret injection — where secrets are retrieved immediately before use and expire after a short TTL — limits the harvesting window. Similarly, AWS IAM roles for GitHub Actions via OIDC federation should replace long-lived access keys; short-lived OIDC tokens cannot be replayed after expiration.

Dependency review for npm and similar package ecosystems should be automated at the registry level using tools that monitor `postinstall` scripts and alert on newly introduced shell execution. According to vendor reporting, Socket.dev [7] and JFrog Artifactory Advanced Security [8] both detected the CanisterWorm `postinstall` pattern within hours of publication. Organizations without automated npm auditing relied on manual discovery, which, based on the incident timeline, in some cases came multiple days after initial deployment.

Strategic Considerations

The TeamPCP campaign illustrates a structural vulnerability in the security toolchain trust model: the tools organizations rely upon to enforce security posture are themselves running with elevated privileges and are rarely subject to the same verification controls applied to production code. Many organizations enforce signature verification for production container images, yet may not apply equivalent verification to the security scanner binaries executed within those pipelines — a gap this attack directly exploited. Closing this gap requires treating security tools as first-class supply chain risk, not as inherently trustworthy by virtue of their function.

The use of an ICP blockchain canister for command-and-control deserves particular attention from security architects. Conventional threat intelligence and infrastructure takedown processes — contacting hosting providers, issuing DMCA or abuse notices, coordinating with registrars — are ineffective against immutable on-chain infrastructure. Defenders must shift from reactive infrastructure takedown to proactive behavior-based detection: monitoring for ICP canister polling patterns in DNS and network flows, and treating any unexplained outbound connection to `.icp0.io` or `.raw.ic0.app` endpoints as suspicious in most enterprise contexts.

Finally, the incomplete credential rotation that enabled this attack suggests a gap in incident response practice that other organizations should examine — one consistent with failure patterns observed in prior supply chain incidents, where partial remediation leaves residual attacker access. Rotating a single credential or token is insufficient when an attacker may have already used the initial access to provision secondary credentials, establish OAuth app authorizations, or register new service accounts. Post-compromise remediation must include a comprehensive audit of all authentication artifacts — including machine identities, service accounts, OAuth applications, and personal access tokens — created or modified during the attacker's access window.

Indicators of Compromise

Type	Indicator	Description
Domain	`scan.aquasecurtiy[.]org`	TeamPCP credential exfiltration (typosquatted)
IP	`45.148.10.212`	Exfil server, TECHOFF SRV LIMITED, Amsterdam

Type	Indicator	Description
Domain	`checkmarx[.]zone`	Checkmarx attack exfiltration domain
IP	`83.142.209.11`	Checkmarx attack exfil server
Domain	`tdtqy-oyaaa-aaaae-af2dq-cai.raw.icp0.io`	ICP canister C2 resolver
Domain	`plug-tab-protective-relay.trycloudflare.com`	GitHub Actions exfiltration channel
IP	`67.217.57[.]240`	December 2025 Sliver C2 infrastructure
File hash (SHA-256)	`822dd269ec10459572dfaaefe163dae693c344249a0161953f0d5cdd110bd2a0`	Malicious Trivy v0.69.4 Linux-64bit
File hash (SHA-256)	`0880819ef821cff918960a39c1c1aada55a5593c61c608ea9215da858a86e349`	Malicious Trivy v0.69.4 Windows-64bit
File hash (SHA-256)	`6328a34b26a63423b555a61f89a6a0525a534e9c88584c815d937910f1ddd538`	Malicious Trivy v0.69.4 macOS-ARM64
Docker image	`aquasec/trivy:0.69.4`, `0.69.5`, `0.69.6`	Malicious Docker Hub releases
File path	`~/.config/systemd/user/sysmon.py`	Trivy stealer persistence dropper
File path	`/var/lib/pgmon/pgmon.py`	CanisterWorm Python backdoor
File path	`/etc/systemd/system/pgmonitor.service`	CanisterWorm persistence service

Type	Indicator	Description
File path	`/tmp/pglog`, `/tmp/.pg_state`	CanisterWorm working artifacts
Kubernetes object	DaemonSet `host-provisioner-iran` in `kube-system`	Iranian-targeted wiper
Kubernetes object	DaemonSet `host-provisioner-std` in `kube-system`	Non-Iranian persistence deployment
GitHub	Repository name prefix `tpcp-docs-`	Defaced Aqua Security repositories
GitHub	Service account `Argon-DevOps-Mgt` (ID 139343333)	Compromised Aqua Security service account

CVEs Referenced

CVE	Description	CVSS	Source
CVE-2026-33634	Trivy GitHub Actions supply chain compromise	9.4 (Critical)	GHSA-69fq-xp46-6x23; Tenable CVE record
CVE-2025-55182	React2Shell — pre-auth RCE in React Server Components	10.0 (Critical)	NVD CVSS v3.x
CVE-2025-29927	React/Next.js middleware bypass enabling RCE	9.1 (Critical)	NVD CVSS v3.1

CSA Resource Alignment

The TeamPCP supply chain attack maps directly to several threat categories addressed in the CSA MAESTRO (Machine Learning Threat Taxonomy and Risk Operations) framework. MAESTRO's analysis of agentic AI systems highlights the risk of compromised tool integrations — CI/CD tools, including security scanners, function as trusted agents within automated pipelines, and their compromise undermines the integrity of every downstream artifact and environment they touch. The attack demonstrates that agentic pipeline trust must be cryptographically grounded, not assumed based on the tool's stated function or reputation.

The CSA Cloud Controls Matrix (CCM) identifies several control domains directly relevant to this incident. The Supply Chain Management, Transparency, and Accountability (STA) domain requires that organizations maintain cryptographic verification for third-party software components integrated into production pipelines — a control that, if applied to GitHub Actions, would have blocked the malicious Trivy tags at the point of consumption. The Identity and Access Management (IAM) domain's controls around least-privilege access and time-limited credential issuance address the broad credential harvesting enabled by long-lived secrets in CI/CD runner memory. The Threat and Vulnerability Management (TVM) domain's requirements for continuous monitoring of third-party component integrity apply directly to the lack of behavioral monitoring that allowed TeamPCP to operate undetected for multiple days across thousands of pipelines.

The CSA Artificial Intelligence and Comprehensive Model (AICM), as a superset of CCM, extends these controls to AI-assisted and autonomous pipeline components. The TeamPCP Stealer's use of AI-assisted code generation — noted by researchers as a defining characteristic of the group's tradecraft — represents precisely the class of threat AICM addresses: attacks that leverage AI acceleration to industrialize known techniques at a speed and scale that outpaces human-operated defenses. Organizations applying AICM controls to their pipeline security posture should treat AI-generated attack tooling as a baseline assumption rather than an advanced threat scenario.

CSA's Zero Trust guidance is directly applicable to the CI/CD context revealed by this attack. The principle of never trusting, always verifying applies as much to automated pipelines as to human users: each workflow step, tool invocation, and artifact download should require cryptographic attestation rather than inheriting ambient trust from the pipeline context. The TeamPCP attack succeeded specifically because pipelines extended implicit trust to a version-tagged artifact — a posture fundamentally incompatible with Zero Trust principles.

References

[1] Aqua Security, "Update: Ongoing Investigation and Continued Remediation," Aqua Security Blog, March 24, 2026. https://www.aquasec.com/blog/trivy-supply-chain-attack-what-you-need-to-know/

[2] Sysdig Threat Research Team, "TeamPCP Expands: Supply Chain Compromise Spreads from Trivy to Checkmarx GitHub Actions," Sysdig Blog, March 23, 2026. https://www.sysdig.com/blog/teampcp-expands-supply-chain-compromise-spreads-from-trivy-to-checkmarx-github-actions

[3] Aikido Security, "TeamPCP Deploys CanisterWorm on NPM Following Trivy Compromise," Aikido Security Blog, March 21, 2026. https://www.aikido.dev/blog/teampcp-deploys-worm-npm-trivy-compromise

[4] Aikido Security, "CanisterWorm Gets Teeth: TeamPCP's Kubernetes Wiper Targets Iran," Aikido Security Blog, March 24, 2026. https://www.aikido.dev/blog/teampcp-stage-payload-canisterworm-iran

[5] Flare.io, "Threat Alert: TeamPCP, An Emerging Force in the Cloud Native and Ransomware Landscape," Flare Blog, February 5, 2026. https://flare.io/learn/resources/blog/teampcp-cloud-native-ransomware

[6] Aqua Security (community discussion), "Trivy Security Incident 2026-03-19," GitHub Discussions, March 19, 2026. https://github.com/aquasecurity/trivy/discussions/10425

[7] Socket.dev, "CanisterWorm: npm Publisher Compromise Deploys Backdoor Across 29+ Packages," Socket.dev Blog, March 21, 2026. https://socket.dev/blog/canisterworm-npm-publisher-compromise-deploys-backdoor-across-29-packages

[8] JFrog Security Research, "New Compromised Versions Detected in CanisterWorm Attack," JFrog Security Research Blog, March 22, 2026. https://research.jfrog.com/post/canister-worm/

Additional Coverage

[9] Wiz, "Trivy Compromised by TeamPCP – Supply Chain Attack Analysis," Wiz Blog, March 22, 2026. https://www.wiz.io/blog/trivy-compromised-teampcp-supply-chain-attack

[10] Rescana, "TeamPCP Worm Targets Docker, Kubernetes, Ray via React2Shell (CVE-2025-55182)," Rescana, February 2026. https://www.rescana.com/post/teampcp-worm-targets-docker-kubernetes-ray-and-redis-via-react2shell-cve-2025-55182-to-build-crim

[11] Security Affairs, "44 Aqua Security Repositories Defaced After Trivy Supply Chain Breach," Security Affairs, March 23, 2026. https://securityaffairs.com/189856/uncategorized/44-aqua-security-repositories-defaced-after-trivy-supply-chain-breach.html

[12] BleepingComputer, "Trivy Vulnerability Scanner Breach Pushed Infostealer via GitHub Actions," BleepingComputer, March 21, 2026. https://www.bleepingcomputer.com/news/security/trivy-vulnerability-scanner-breach-pushed-infostealer-via-github-actions/

[13] BleepingComputer, "TeamPCP Deploys Iran-Targeted Wiper in Kubernetes Attacks," BleepingComputer, March 24, 2026. https://www.bleepingcomputer.com/news/security/teampcp-deploys-iran-targeted-wiper-in-kubernetes-attacks/

[14] The Hacker News, "Trivy Hack Spreads Infostealer via Docker, Triggers Worm and Kubernetes Wiper," The Hacker News, March 24, 2026. https://thehackernews.com/2026/03/trivy-hack-spreads-infostealer-via.html

[15] The Hacker News, "Trivy Supply Chain Attack Triggers Self-Spreading CanisterWorm Across 47 npm Packages," The Hacker News, March 21, 2026. https://thehackernews.com/2026/03/trivy-supply-chain-attack-triggers-self.html

[16] KrebsOnSecurity, "'CanisterWorm' Springs Wiper Attack Targeting Iran," KrebsOnSecurity, March 24, 2026. https://krebsonsecurity.com/2026/03/canisterworm-springs-wiper-attack-targeting-iran/

[17] phoenix.security, "Trivy Supply Chain Compromise – TeamPCP Weaponised Scanner Ongoing Attack," Phoenix Security, March 22, 2026. https://phoenix.security/trivy-supply-chain-compromise-teampcp-weaponised-scanner-ongoing-attack/