INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-30 06:30 UTC

Vect 2.0 Ransomware Destroys Data It Claims to Encrypt: Why Paying Won't Help

THREAT CAMPAIGN | **CRITICAL** | CVSS 9.5

SCC Item ID	SCC-CAM-2026-0241
Type	Threat Campaign
Severity	CRITICAL
CVSS Base Score	9.5
Affected Products	Organizations compromised via TeamPCP supply chain attack; confirmed downstream targets include Trivy, Checkmarx, KICS, and LiteLLM ecosystems, specific vendor product versions not fully identified in available sources
Published	2026-04-29T11:23:53
Discovery Source	Rss

Executive Summary

A ransomware variant called Vect 2.0, deployed through a supply chain attack targeting widely used open-source security tools, destroys data rather than encrypting it, meaning ransom payment will not recover lost files. Organizations using Trivy, Checkmarx, KICS, or LiteLLM in their development or security pipelines are potentially affected. The business risk is permanent data loss, extended operational outage, and the compounding exposure of having security tooling itself serve as the attack vector. Specific affected package versions are pending full vendor disclosure; organizations using these tools should begin dependency integrity audits immediately while awaiting version-specific indicators.

Technical Analysis

Vect 2.0 is a ransomware payload deployed as the final stage of the TeamPCP multi-stage supply chain campaign. TeamPCP compromised packages or components within the Trivy vulnerability scanner, Checkmarx, KICS (Keeping Infrastructure as Code Secure), and LiteLLM ecosystems. A design flaw in Vect 2.0's implementation renders its encryption nonfunctional; data is overwritten or destroyed rather than reversibly encrypted, making the threat actor's decryptor inoperable. This reclassifies Vect 2.0 as a wiper (CWE-693: Protection Mechanism Failure; CWE-311: Missing Encryption of Sensitive Data used destructively). Relevant MITRE ATT&CK techniques: T1195.001 and T1195.002 (Supply Chain Compromise, software and dependency poisoning), T1486 (Data Encrypted for Impact, functionally a wiper), T1485 (Data Destruction), T1489 (Service Stop), T1490 (Inhibit System Recovery), T1059 (Command and Scripting Interpreter), T1071 (Application Layer Protocol for C2). Specific affected package versions and full IOC sets are pending confirmation from primary

vendor disclosures. No CVE identifier is assigned. Cross-reference indicators directly with Unit 42, Arctic Wolf, and Wiz threat reports for authoritative IOC sets.

Action Checklist

1. Step 1: Containment. Immediately isolate any system where Trivy, Checkmarx, KICS, or LiteLLM components were recently installed or updated, particularly in CI/CD pipelines. Disconnect affected hosts from the network. Do not reboot; preserve volatile memory for forensic triage. Suspend automated pipelines consuming these tools until dependency integrity is confirmed.
2. Step 2: Detection. Audit software bill of materials (SBOM) and dependency manifests for Trivy, Checkmarx, KICS, and LiteLLM packages installed within the campaign's active window. Check pipeline logs for unexpected script execution (T1059), outbound C2 connections (T1071), or file overwrite activity at scale. Review endpoint and EDR telemetry for mass file modification events inconsistent with normal build activity. Unit 42, Arctic Wolf, and Wiz published campaign-specific IOCs; cross-reference those reports directly for hashes and infrastructure indicators.
3. Step 3: Eradication. Remove and quarantine any confirmed compromised packages. Roll back to known-good dependency versions verified against official upstream checksums. Re-validate the integrity of your CI/CD pipeline toolchain from the build environment outward. Do not trust cached or locally stored copies of affected packages.
4. Step 4: Recovery. Before committing to permanent loss classification, engage incident response and forensic specialists to confirm data destruction versus encrypted-but-unrecoverable state; this determination affects business continuity and regulatory notification timelines. If data is confirmed destroyed, initiate disaster recovery from the most recent verified clean backup predating the compromise window. Validate backup integrity before restore. After restoration, re-verify all pipeline dependencies before returning to production. Monitor restored systems for re-infection indicators.
5. Step 5: Post-Incident. Conduct a dependency trust audit across all open-source tooling in security and development pipelines. Implement supply chain integrity controls: checksum verification, signed package enforcement, and SBOM generation at build time. Map gaps against NIST SP 800-161 (Supply Chain Risk Management) and CISA's Secure Software Development guidance. Review incident detection coverage for wiper-class behavior; many ransomware playbooks assume recoverability and will fail against wipers.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to executive leadership, legal counsel, and external IR retainer immediately if any of the following are confirmed: (1) Vect 2.0 payload executed on systems with access to production secrets, signing keys, or customer PII/PHI — triggering breach notification obligations under GDPR, HIPAA, or applicable state law; (2) compromised pipeline artifacts were published to downstream customers or internal consumers before containment; (3) the affected CI/CD environment has write access to production infrastructure, creating a potential secondary compromise vector beyond the initial data destruction.

Recovery Notes	Because Vect 2.0 destroys rather than encrypts data, recovery time is entirely dependent on backup recency and integrity — there is no decryption path and no negotiation option that changes this outcome. After restoring from verified clean backups, maintain elevated monitoring on restored CI/CD systems for a minimum of 30 days, specifically watching for any re-introduction of the affected package versions (Trivy, Checkmarx, KICS, LiteLLM) via developer workstations, cached artifacts, or automated dependency update bots such as Dependabot or Renovate. Validate that all pipeline secrets (API keys, signing certificates, deployment credentials) exposed on compromised systems have been rotated before returning any restored pipeline to production, as the TeamPCP supply chain vector may have exfiltrated credentials as a secondary objective alongside the destructive payload.
Forensic Artifacts	CI/CD job execution logs from Jenkins ($JENKINS_HOME/jobs//builds//log), GitHub Actions workflow run archives, or GitLab job traces — these will show the exact timestamp and subprocess tree of the malicious package install hook execution for Trivy, Checkmarx, KICS, or LiteLLM during the TeamPCP campaign window | pip install logs (~/.pip/pip.log, /root/.pip/pip.log) and the pip HTTP cache (~/.cache/pip/http/) containing the downloaded wheel or sdist for the malicious LiteLLM package version, preserving the source registry URL and download hash for comparison against PyPI-published legitimate digests | Python dist-info RECORD files (located at site-packages/-.dist-info/RECORD) for the four affected packages — the RECORD file lists every file installed and its expected hash; any files added or modified by the wiper payload post-install will appear as additions or hash mismatches against this manifest | Sysmon Event ID 11 (FileCreate) and Event ID 23 (FileDelete) logs, or Linux inotify/auditd logs with `-w -p war` rules, capturing the mass file overwrite or zero-fill pattern that is the forensic signature of Vect 2.0's data destruction phase — the volume and rate of these events (expected: hundreds to thousands of events per minute across source code and artifact directories) distinguishes wiper activity from normal build I/O | Network flow records or tcpdump packet captures from CI/CD runner network interfaces covering the compromise window — specifically capturing any outbound connections from the build process tree to non-RFC1918 addresses on uncommon ports, corresponding to MITRE T1071 (Application Layer Protocol) C2 behavior attributed to the TeamPCP campaign infrastructure in the Arctic Wolf, Wiz, and Unit 42 reporting

Per-Action IR Details

Step 1: Containment — Immediately isolate any system where Trivy, Checkmarx, KICS, or LiteLLM components were recently installed or updated, particularly in CI/CD pipelines. Disconnect affected hosts from the network. Do not reboot — preserve volatile memory for forensic triage. Suspend automated pipelines consuming these tools until dependency integrity is confirmed.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment Strategy

Controls: NIST IR-4 (Incident Handling), NIST SI-3 (Malicious Code Protection), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices)

Compensating: On Linux CI/CD runners, immediately run `ip link set down` or `iptables -I INPUT -j DROP && iptables -I OUTPUT -j DROP` to hard-isolate without reboot. On Windows build agents, use `netsh advfirewall set allprofiles firewallpolicy blockinbound,blockoutbound`. Capture volatile memory first using Avast's free WinPmem (Windows) or `sudo avml /tmp/mem.lime` with the AVML tool (Linux) before any network changes if forensic capability exists. For pipeline suspension, revoke CI/CD service account tokens in GitHub Actions, GitLab, or Jenkins immediately — do not merely disable the job, as queued runs may still execute.

Evidence: Before isolating: capture a full process list (`ps auxf` on Linux, `Get-Process | Select-Object Name,Id,Path,CommandLine` on Windows) to identify any child processes spawned by trivy, checkmarx-ast-cli, kics, or litellm package scripts. Capture active network connections via `ss -tulnp` or `netstat -anob` to identify any live C2

sessions originating from these processes. On Linux, dump `/proc//maps` and `/proc//fd` for any suspicious PIDs. On Windows, capture the Prefetch directory (`C:\Windows\Prefetch`) and the AmCache hive (`C:\Windows\appcompat\Programs\Amcache.hve`) — these will show recently executed binaries dropped by malicious package install hooks before the reboot wipes volatile state.

Step 2: Detection — Audit software bill of materials (SBOM) and dependency manifests for Trivy, Checkmarx, KICS, and LiteLLM packages installed within the campaign's active window. Check pipeline logs for unexpected script execution (T1059), outbound C2 connections (T1071), or file overwrite activity at scale. Review endpoint and EDR telemetry for mass file modification events inconsistent with normal build activity. Arctic Wolf, Wiz, and Unit 42 published campaign-specific IOCs — cross-reference those reports directly for hashes and infrastructure indicators.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST SI-4 (System Monitoring), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-2 (Event Logging), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 8.2 (Collect Audit Logs)

Compensating: Without EDR, deploy Sysmon with a configuration tuned for supply chain abuse: specifically enable Event ID 1 (Process Create) to catch post-install hook execution from pip, npm, or Go module install scripts, and Event ID 11 (File Create) to detect mass file overwrite patterns consistent with Vect 2.0's destruction behavior. Use osquery with the query `SELECT name, version, install_time FROM python_packages WHERE name IN ('litellm','trivy','kics')` to enumerate installed package versions across the fleet. For SBOM diffing without tooling, run `pip show litellm` and compare the `Location` field checksum against the official PyPI SHA256 digest using `pip hash --algorithm sha256 `. For C2 detection without SIEM, run `tcpdump -i any -w /tmp/ci_capture.pcap` on CI runners during build execution and inspect in Wireshark filtering on non-RFC1918 destinations from the build process tree. Apply the public Sigma rule `proc_creation_win_susp_script_exec_from_env` adapted for package manager child processes as a detection signal for T1059 execution via install hooks.

Evidence: Retrieve pip install logs from `~/.pip/pip.log` or `/root/.pip/pip.log` and the pip HTTP cache at `~/.cache/pip/` — malicious package versions will have been fetched from PyPI or a typosquatted registry and the download URL will appear in these logs. For Node-based tooling, inspect `~/.npm/_logs/` for install hook execution. On CI systems, pull the full job execution logs from Jenkins (`$JENKINS_HOME/jobs//builds//log`), GitHub Actions workflow run logs, or GitLab job trace artifacts — look for unexpected subprocess invocations (bash, sh, python, curl, wget) spawned during `pip install`, `npm install`, or `go get` for the four affected packages. Cross-reference package hashes in your lock files (requirements.txt, package-lock.json, go.sum) against the IOC hashes published by Arctic Wolf, Wiz, and Unit 42 for the TeamPCP campaign.

Step 3: Eradication — Remove and quarantine any confirmed compromised packages. Roll back to known-good dependency versions verified against official upstream checksums. Re-validate the integrity of your CI/CD pipeline toolchain from the build environment outward. Do not trust cached or locally stored copies of affected packages.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CM-3 (Configuration Change Control) — implied via NIST 800-53r5 CM family, CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 2.3 (Address Unauthorized Software), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management)

Compensating: For pip-managed packages (LiteLLM), run `pip uninstall litellm -y && pip cache purge` then reinstall only after verifying the target version's SHA256 against PyPI's JSON API: `curl https://pypi.org/pypi/litellm//json | python3 -m json.tool | grep sha256`. For container-based Trivy or KICS deployments, do not `docker pull` — instead pull the specific digest-pinned image (`docker pull aquasec/trivy@sha256:`) from the official registry and verify against the vendor's published digest. Purge all local pip, npm, and Go module caches on every build agent: `pip cache purge`, `npm cache clean --force`, `go clean -modcache`. Re-pin all four affected package references in lock files to verified-good versions before re-enabling pipelines. Use `sha256sum` or `certutil -hashfile` to generate and store

checksums of the verified-clean binaries as a baseline for future integrity checks.

Evidence: Before removing packages, preserve forensic copies: archive the full site-packages directory for Python (`tar czf /forensics/site-packages-$(hostname)-$(date +%s).tgz $(python3 -m site --user-site)`) and the global packages dir (`pip show litellm | grep Location`). Preserve the package's dist-info directory which contains the RECORD file listing all installed files and their hashes — this will show if files were added or modified post-install by the malicious payload. Capture any persistence mechanisms written during package install: on Linux check `/etc/cron.d/`, `/etc/profile.d/`, `~/.bashrc`, `~/.profile`, and systemd unit files in `/etc/systemd/system/` for entries created within the compromise window. On Windows, check `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run` and scheduled tasks via `schtasks /query /fo LIST /v` for entries created by package install hooks.

Step 4: Recovery — Do not attempt data recovery from Vect 2.0-affected systems through decryption; the data is destroyed, not encrypted. Initiate disaster recovery from the most recent verified clean backup predating the compromise window. Validate backup integrity before restore. After restoration, re-verify all pipeline dependencies before returning to production. Monitor restored systems for re-infection indicators.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-11 (Audit Record Retention), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 3.4 (Enforce Data Retention)

Compensating: To establish the compromise window boundary for backup selection, correlate the earliest malicious package install timestamp from pip logs or CI job logs against your backup schedule — select the most recent backup with a completion timestamp strictly before that install event. Before restoring, verify backup integrity using the backup system's built-in checksum or by running `sha256sum` against archived files and comparing to a stored manifest. After restore, use AIDE (Advanced Intrusion Detection Environment) or Tripwire Free Edition to establish a new filesystem baseline on the restored system: `aide --init && mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db`. For re-infection monitoring post-restore, deploy a Sysmon rule specifically watching for write access to source code directories, CI configuration files (`.github/workflows/`, `Jenkinsfile`, `.gitlab-ci.yml`), and dependency lock files, which are the Vect 2.0 campaign's likely persistence targets.

Evidence: Before restoring, document the full extent of Vect 2.0 file destruction: run `find / -newer /tmp/compromise_timestamp_reference_file -type f -name '*.vect2'` or search for files with zero-byte content or overwritten headers inconsistent with file type (use `file` command on a sample). Capture a disk image of at least one confirmed-destroyed system using `dc3dd if=/dev/sda | gzip > /forensics/destroyed_host.img.gz` before reimaging — this preserves evidence of the wiper's file targeting logic for post-incident analysis. Document which specific pipeline artifacts, build outputs, source checkouts, secrets, or configuration files were in scope on affected systems, as this determines downstream breach notification obligations.

Step 5: Post-Incident — Conduct a dependency trust audit across all open-source tooling in security and development pipelines. Implement supply chain integrity controls: checksum verification, signed package enforcement, and SBOM generation at build time. Map gaps against NIST SP 800-161 (Supply Chain Risk Management) and CISA's Secure Software Development guidance. Review incident detection coverage for wiper-class behavior — many ransomware playbooks assume recoverability and will fail against wipers.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST SI-7 (Software, Firmware, and Information Integrity), NIST SI-2 (Flaw Remediation), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Generate SBOMs for all CI/CD pipelines using Syft (free, Anchore): `syft dir:. -o cyclonedx-json > sbom.json` — commit this output as a build artifact and diff it on each subsequent build to detect unexpected dependency additions. Enforce checksum pinning in requirements files: use `pip-compile --generate-hashes` to produce a fully hash-pinned requirements.txt for all Python tooling including LiteLLM. For signed package enforcement without enterprise tooling, configure pip to use `--require-hashes` mode by adding `require-hashes = true` to `pip.conf`.

To close the wiper detection gap identified in this incident, write a Sigma rule targeting mass file modification events (>100 file write events in <60 seconds from a single process) and test it against your log pipeline using `sigma convert` — this class of behavior is typically absent from ransomware-oriented playbooks but is the defining forensic signature of Vect 2.0's destructive payload.

Evidence: Produce a lessons-learned artifact that documents: (1) the earliest evidence timestamp of the malicious TeamPCP package in your environment derived from pip logs, CI logs, and artifact repository pull records; (2) a complete list of systems, pipelines, and data assets confirmed or suspected to have loaded the compromised Trivy, Checkmarx, KICS, or LiteLLM package; (3) detection gaps — specifically whether your SIEM, EDR, or pipeline monitoring generated any alert on the mass file modification or outbound C2 behavior, and the delta between compromise time and detection time. This timeline document is required for regulatory breach notification analysis and for updating your IR plan's wiper-class incident procedures.

Detection Guidance

Detection should focus on three layers. First, dependency integrity: compare installed package hashes for Trivy, Checkmarx, KICS, and LiteLLM against upstream official checksums. Any mismatch warrants immediate investigation. Second, behavioral indicators of wiper activity: look for high-volume file modification or deletion events in short time windows (EDR telemetry, file integrity monitoring alerts), service termination events preceding file activity (T1489/T1490), and inhibition of VSS shadow copies or backup services. Third, C2 and lateral movement: review network logs for anomalous outbound connections from build servers or pipeline runners, particularly to infrastructure not consistent with normal dependency resolution. Cross-reference published IOCs from Unit 42 (unit42.paloaltonetworks.com/teampcp-supply-chain-attacks), Arctic Wolf (arcticwolf.com/resources/blog/teampcp-supply-chain-attack-campaign-targets-trivy-checkmarx-kics-and-litellm-potential-downstream-impact-to-additional-projects), and Wiz (wiz.io/blog/trivy-compromised-teampcp-supply-chain-attack) for campaign-specific indicators. Validate all URLs before use and prioritize indicators from these primary sources.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`See Unit 42 report: unit42.paloaltonetworks.com/teampcp-supply-chain-attacks/`	Campaign-specific IOCs for TeamPCP infrastructure published by Palo Alto Unit 42 — hashes, domains, and IPs not reproduced here to avoid transcription error; retrieve directly from source	MEDIUM
URL	`See Arctic Wolf report: arcticwolf.com/resources/blog/teampcp-supply-chain-attack-campaign-targets-trivy-checkmarx-kics-and-litellm-potential-downstream-impact-to-additional-projects/`	Arctic Wolf campaign analysis with additional IOCs and affected package details	MEDIUM
URL	`See Wiz blog: wiz.io/blog/trivy-compromised-teampcp-supply-chain-attack`	Wiz analysis focused on Trivy compromise specifics within the TeamPCP campaign	MEDIUM

Framework Mappings

MITRE-ATTACK

- **T1486** — Data Encrypted for Impact
- **T1489** — Service Stop
- **T1071** — Application Layer Protocol
- **T1195.001** — Compromise Software Dependencies and Development Tools
- **T1485** — Data Destruction
- **T1490** — Inhibit System Recovery
- **T1059** — Command and Scripting Interpreter
- **T1195.002** — Compromise Software Supply Chain

NIST-800-53R5

- **CP-9** — System Backup
- **CP-10** — System Recovery and Reconstitution
- **CM-6** — Configuration Settings
- **SI-4** — System Monitoring
- **CA-7** — Continuous Monitoring
- **SC-7** — Boundary Protection
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-7** — Software, Firmware, and Information Integrity
- **SA-9** — External System Services
- **SR-3** — Supply Chain Controls and Processes
- **IR-4** — Incident Handling
- **SR-2** — Supply Chain Risk Management Plan
- **SC-13** — Cryptographic Protection

NIST-CSF-2

- **RS.MI-01** — Incidents are contained
- **GV.SC-01** — Cybersecurity supply chain risk management program

HIPAA-SECURITY

- **164.308(a)(7)(ii)(A)** — Data Backup Plan
- **164.312(e)(1)** — Transmission Security

ISO-27001-2022

- **A.5.29** — Information security during disruption
- **A.8.8** — Management of technical vulnerabilities
- **A.5.21** — Managing information security in the ICT supply chain
- **A.8.24** — Use of cryptography

CIS-V8

- **15.1** — Establish and Maintain an Inventory of Service Providers

SOC2-TSC

- **CC9.2** — Manages risks associated with vendors and business partners

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1486	Data Encrypted for Impact	Impact
T1489	Service Stop	Impact
T1071	Application Layer Protocol	Command-And-Control
T1195.001	Compromise Software Dependencies and Development Tools	Initial-Access
T1485	Data Destruction	Impact
T1490	Inhibit System Recovery	Impact
T1059	Command and Scripting Interpreter	Execution
T1195.002	Compromise Software Supply Chain	Initial-Access

Sources

Source	URL	Tier
Security News	https://www.darkreading.com/threat-intelligence/vect-ransomware-wip...	T3
TeamPCP's Multi-Stage Supply Chain Attack on Security Infrastructure	https://unit42.paloaltonetworks.com/teampcp-supply-chain-attacks/	T3
TeamPCP Supply Chain Attack Campaign Targets Trivy, Checkmarx ...	https://arcticwolf.com/resources/blog/teampcp-supply-chain-attack-c...	T3
Trivy Compromised by "TeamPCP" | Wiz Blog	https://www.wiz.io/blog/trivy-compromised-teampcp-supply-chain-attack	T3
Inside the TeamPCP Supply Chain Attack | SANS Institute	https://www.sans.org/webcasts/when-security-scanner-became-weapon	T3

DISCLAIMER

This intelligence report is produced by Tech Jacks Solutions Security Command Center (SCC) for informational purposes only. It does not constitute professional security advice, legal counsel, or an incident response engagement. The information herein is derived from publicly available sources and AI-assisted analysis; while every effort is made to ensure accuracy, Tech Jacks Solutions makes no warranties regarding completeness or timeliness. Organizations should conduct their own validation and consult qualified security professionals before taking action based on this report. Tech Jacks Solutions is not liable for any damages resulting from the use of this information.

Generated 2026-04-30 06:30 UTC by TJS Security Command Center