INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-12 19:06 UTC

TeamPCP Turns Checkmarx's Own Credentials Against Jenkins Users in Third Supply-Chain Strike

THREAT CAMPAIGN | **CRITICAL** | CVSS 9.5

SCC Item ID	SCC-CAM-2026-0306
Type	Threat Campaign
Severity	CRITICAL
CVSS Base Score	9.5
Affected Products	Checkmarx Jenkins AST Plugin v2026.5.09 (malicious); Jenkins Marketplace; Checkmarx KICS; Checkmarx Trivy integration; Docker; VSCode; Open VSX
Published	2026-05-11T18:03:06
Discovery Source	Rss

Executive Summary

A threat actor known as TeamPCP used credentials stolen in a prior Checkmarx breach to publish a malicious version of the official Checkmarx Jenkins AST Plugin (v2026.5.09) to the Jenkins Marketplace on May 9, 2026. Any organization that installed this version has had credential-stealing malware injected directly into their CI/CD pipeline, placing all pipeline secrets, tokens, and access keys at immediate risk of exfiltration. This is the third Checkmarx artifact compromised in a coordinated campaign spanning multiple months, enabled each time by Checkmarx's failure to rotate credentials after the initial breach.

Technical Analysis

TeamPCP published a trojanized build of the Checkmarx Jenkins AST Plugin (v2026.5.09) to the official Jenkins Marketplace on May 9, 2026. The malicious plugin delivers credential-stealing malware at installation time, targeting secrets stored in Jenkins CI/CD pipelines. The attack chain exploits credentials obtained during a March 2026 compromise of Checkmarx's Trivy integration, credentials that Checkmarx failed to rotate after the initial incident, enabling lateral movement across Checkmarx-managed artifact publication channels. This is the third distinct artifact compromised in the same campaign. The attack surface includes Jenkins-integrated CI/CD environments, Docker container workflows, and editor ecosystems (VSCode, Open VSX). Relevant CWEs: CWE-494 (Download of Code Without Integrity Check), CWE-798 (Use of Hard-coded Credentials), CWE-522 (Insufficiently Protected Credentials), CWE-693 (Protection Mechanism Failure). MITRE ATT&CK techniques include T1195.001 (Compromise Software Supply Chain), T1199 (Trusted Relationship), T1608.001 (Stage

Capabilities: Upload Malware), T1552.001 (Credentials In Files), T1078 (Valid Accounts), T1176 (Browser Extensions, applicable to VSCode/Open VSX vector), T1059 (Command and Scripting Interpreter), T1574 (Hijack Execution Flow), and T1588.003 (Obtain Capabilities: Code Signing Certificates). No CVE has been assigned. No CVSS vendor score is available from the vendor advisory. The source-assigned CVSS base is 9.5. No CISA KEV entry exists at this time.

Action Checklist

1. Step 1: Containment, Immediately identify any Jenkins instances in your environment that installed Checkmarx Jenkins AST Plugin v2026.5.09. Remove the plugin from all affected Jenkins nodes without delay. Block v2026.5.09 at your artifact proxy or package manager (e.g., Nexus, Artifactory) to prevent reinstallation. Isolate affected build agents from production networks pending full credential rotation. Reference Checkmarx's May 9 incident update at https://checkmarx.com/blog/supply-chain-security-incident-update-may-9/ for vendor-specific guidance.
2. Step 2: Detection, Audit Jenkins plugin installation logs for any record of v2026.5.09 being installed, particularly between May 9, 2026 and discovery. Review Jenkins system logs and pipeline execution logs for anomalous outbound connections or unexpected process spawning during or after plugin installation. Search pipeline logs for credential access events that do not correspond to expected job activity. Check for unauthorized use of pipeline secrets, tokens, or service account credentials in downstream systems. If a SIEM is present, query for process execution or network events originating from Jenkins build agents to external IPs during the exposure window.
3. Step 3: Eradication, Remove Checkmarx Jenkins AST Plugin v2026.5.09 from all Jenkins instances. Do not replace with any Checkmarx Jenkins plugin version until Checkmarx explicitly confirms a clean, verified replacement build and provides integrity verification (hash or signature). Rotate all credentials, tokens, API keys, and secrets that were accessible to Jenkins pipelines on affected nodes; treat every secret as compromised. This includes cloud provider keys, container registry tokens, code signing credentials, and any secrets stored in Jenkins credential stores or pipeline environment variables. Audit exposure of Checkmarx KICS and Checkmarx Trivy integration artifacts per earlier compromise advisories.
4. Step 4: Recovery, After credential rotation, verify no unauthorized access occurred against downstream systems using the rotated secrets (check cloud provider access logs, container registry audit logs, source control audit logs). Re-enable affected build pipelines only after confirming plugin removal and full secret rotation. Implement integrity verification (checksum or signature validation) for all reinstalled plugins before restoring production pipeline activity. Monitor pipeline behavior for anomalous outbound connections or unexpected process execution for at least 30 days post-remediation.
5. Step 5: Post-Incident, This attack succeeded because Checkmarx did not rotate publication credentials after its March 2026 Trivy compromise. Internally, review your own credential rotation policy following any third-party supply chain incident affecting tools in your pipeline. Evaluate whether your software supply chain controls include integrity verification (signed artifacts, hash validation) for third-party plugins installed into CI/CD systems. Consider restricting Jenkins plugin installation to an approved, internally mirrored repository where artifacts are scanned before acceptance. Map this incident to NIST SP 800-161r1 (C-SCRM) controls for third-party software integrity assurance.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to CISO, legal, and external IR retainer immediately if cloud provider logs, container registry audit logs, or source control audit logs confirm any use of Jenkins pipeline secrets between May 9, 2026 and credential rotation — particularly if those credentials had access to production systems, customer data environments, or code signing infrastructure, as this constitutes a confirmed supply chain breach with potential regulatory notification obligations under GDPR, CCPA, or sector-specific breach notification requirements.
Recovery Notes	Before re-enabling any build pipeline, confirm: (1) all credentials accessible to the affected Jenkins instances have been rotated and revoked — not just the secrets explicitly named in Jenkinsfiles, but any credential reachable by the Jenkins service account at the OS or cloud provider IAM level; (2) the replacement Checkmarx Jenkins plugin version has been hash-verified against Checkmarx's official post-incident advisory and installed only from a controlled internal mirror; and (3) downstream container images built during the exposure window (May 9 through remediation) have been inventoried and assessed for potential tampering before further promotion through the pipeline. Maintain elevated monitoring of Jenkins build agent outbound network activity, Jenkins credential store access events, and cloud provider API calls from pipeline service accounts for a minimum of 30 days, as TeamPCP's three-incident pattern suggests persistent access objectives rather than opportunistic credential theft.
Forensic Artifacts	Checkmarx Jenkins AST Plugin v2026.5.09 binary: $JENKINS_HOME/plugins/checkmarx.jpi — preserve SHA-256 hash immediately; this is the primary IOC and should be submitted to Checkmarx PSIRT and shared with threat intel partners via STIX/TAXII or direct disclosure | Jenkins build executor logs for all jobs that ran between May 9, 2026 and remediation: $JENKINS_HOME/jobs/*/builds/*/log — the malicious plugin's credential exfiltration would appear as anomalous outbound HTTP POST requests or DNS lookups to TeamPCP infrastructure embedded within normal build output | Jenkins master key and credentials.xml access timestamps: run 'stat $JENKINS_HOME/secrets/master.key $JENKINS_HOME/credentials.xml $JENKINS_HOME/secrets/hudson.util.Secret' — if last-accessed timestamps fall within the exposure window during non-build periods, this indicates direct credential store access by the malicious plugin | OS-level network connection logs from Jenkins build agents scoped to the exposure window: on Linux, /var/log/audit/audit.log entries with syscall=connect and uid matching the jenkins service account; on Windows, Sysmon Event ID 3 (Network Connection) with Image path matching java.exe — external destination IPs are candidate TeamPCP C2 or exfiltration endpoints | Cloud provider IAM and secrets manager audit logs for all service accounts and access keys accessible to affected pipelines: AWS CloudTrail GetSecretValue/AssumeRole events, GCP Cloud Audit Logs for iam.serviceAccounts.actAs, and Azure Activity Log for Key Vault secret access — scoped to May 9, 2026 through key rotation timestamp to identify whether TeamPCP used exfiltrated credentials against downstream infrastructure

Per-Action IR Details

Step 1: Containment — Immediately identify any Jenkins instances in your environment that installed Checkmarx Jenkins AST Plugin v2026.5.09. Remove the plugin from all affected Jenkins nodes without delay. Block v2026.5.09 at your artifact proxy or package manager (e.g., Nexus, Artifactory) to prevent reinstallation. Isolate affected build agents from production networks pending full credential rotation. Reference Checkmarx's May 9 incident update at https://checkmarx.com/blog/supply-chain-security-incident-update-may-9/ for vendor-specific guidance.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST CM-7 (Least Functionality), NIST SI-7 (Software, Firmware, and Information Integrity), CIS 2.3 (Address Unauthorized Software), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: Run `java -jar jenkins-cli.jar -s http://localhost:8080/ list-plugins | grep -i checkmarx` on each Jenkins controller to enumerate installed plugin versions. In Jenkins itself, navigate to Manage Jenkins → Plugins → Installed and filter for 'checkmarx' — document the version string before removal. To block at Artifactory, add v2026.5.09 to a block list under Admin → Repositories → Remote → Exclude Patterns using `*checkmarx-ast-plugin-2026.5.09*`. On build agents without firewall tooling, use `iptables -I OUTPUT -m owner --uid-owner jenkins -j DROP` as a temporary network isolation measure to prevent further exfiltration from the Jenkins service account while remediation proceeds.

Evidence: Before removing the plugin, capture the Jenkins plugin directory state: on Linux, snapshot `$JENKINS_HOME/plugins/checkmarx*.jpi` and `*.jpi.pinned` files including file hashes (`sha256sum $JENKINS_HOME/plugins/checkmarx*.jpi`). Preserve the full Jenkins system log (`$JENKINS_HOME/logs/jenkins.log`) and all build executor logs from the exposure window (May 9, 2026 through discovery date) found under `$JENKINS_HOME/jobs/*/builds/*/log`. Capture active outbound network connections from the Jenkins process at time of discovery using `ss -tunp | grep java` or `netstat -tunp | grep java` to identify any live exfiltration channels before isolating the node. Preserve these artifacts to immutable storage before proceeding.

Step 2: Detection — Audit Jenkins plugin installation logs for any record of v2026.5.09 being installed, particularly between May 9, 2026 and discovery. Review Jenkins system logs and pipeline execution logs for anomalous outbound connections or unexpected process spawning during or after plugin installation. Search pipeline logs for credential access events that do not correspond to expected job activity. Check for unauthorized use of pipeline secrets, tokens, or service account credentials in downstream systems. If a SIEM is present, query for process execution or network events originating from Jenkins build agents to external IPs during the exposure window.

NIST Phase: Detection Analysis

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

Controls: NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), NIST SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Without a SIEM, execute the following targeted searches on the Jenkins controller: (1) Parse plugin installation history from `$JENKINS_HOME/logs/jenkins.log` using `grep -E '(checkmarx|2026\.5\.09|plugin.*install)' $JENKINS_HOME/logs/jenkins.log`. (2) Detect anomalous process spawning from the Jenkins JVM using Sysmon Event ID 1 (Process Creation) — look for `java.exe` or `jenkins.war` as parent process spawning `cmd.exe`, `sh`, `curl`, `wget`, or `powershell.exe`. Deploy this Sysmon config rule targeting the jenkins service account UID on Linux build agents. (3) Use `ausearch -ua jenkins --start 05/09/2026 --end today` on auditd-enabled Linux agents to pull all syscalls made by the Jenkins process user. (4) For outbound connection detection without a SIEM, run `grep -E '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}' $JENKINS_HOME/jobs/*/builds/*/log | grep -v RFC1918` to surface external IPs contacted during pipeline execution. (5) Cross-reference any identified external IPs against TeamPCP-associated infrastructure using free threat intel sources (VirusTotal, AbuseIPDB, Shodan).

Evidence: The malicious plugin's credential-stealing mechanism would produce specific artifacts: (1) In Jenkins build logs (`$JENKINS_HOME/jobs/*/builds/*/log`), look for unexpected HTTP POST or DNS requests to non-Checkmarx external endpoints during plugin initialization or first pipeline execution after May 9. (2) In Jenkins credential store access logs, the plugin would need to call `com.cloudbees.plugins.credentials` APIs — on systems with audit logging enabled, these appear in the Jenkins audit trail plugin log or in JVM debug output if `-Djava.util.logging.config.file` is configured. (3) On Linux build agents, `/proc//net/tcp` and `/proc//net/tcp6` snapshots reveal open connections at time of discovery. (4) Review `$JENKINS_HOME/secrets/` directory access timestamps using `stat` or `find $JENKINS_HOME/secrets -newer /tmp/timestamp_marker` to detect if the plugin read master keys or credential blobs. (5) Check OS-level DNS resolution logs or `/etc/hosts` modifications on build agents for new entries introduced post-plugin-install.

Step 3: Eradication — Remove Checkmarx Jenkins AST Plugin v2026.5.09 from all Jenkins instances. Do not replace with any Checkmarx Jenkins plugin version until Checkmarx explicitly confirms a clean, verified replacement build and provides integrity verification (hash or signature). Rotate all credentials, tokens, API keys, and secrets that were accessible to Jenkins pipelines on affected nodes — treat every secret as compromised. This includes cloud provider keys, container registry tokens, code signing credentials, and any secrets stored in Jenkins credential stores or pipeline environment variables. Also audit exposure of Checkmarx KICS and Checkmarx Trivy integration artifacts per earlier compromise advisories.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST IR-4 (Incident Handling), NIST SI-2 (Flaw Remediation), NIST IA-5 (Authenticator Management), NIST CM-3 (Configuration Change Control), CIS 5.2 (Use Unique Passwords), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Plugin removal: `java -jar jenkins-cli.jar -s http://localhost:8080/ disable-plugin checkmarx` followed by manual deletion of `$JENKINS_HOME/plugins/checkmarx.jpi` and `$JENKINS_HOME/plugins/checkmarx/` directory; restart Jenkins to confirm removal. For credential rotation without a secrets management platform, enumerate all credentials in the Jenkins credential store via the Jenkins Script Console (Manage Jenkins → Script Console): `com.cloudbees.plugins.credentials.CredentialsProvider.lookupCredentials(com.cloudbees.plugins.credentials.Credentials.class, Jenkins.instance, null, null).each { println it.id + ' : ' + it.class }` — document every credential ID, then rotate each externally before re-entering. For pipeline environment variables, grep all Jenkinsfiles and shared library groovy files: `find $JENKINS_HOME -name 'Jenkinsfile' -o -name '*.groovy' | xargs grep -l 'credentials\|withCredentials\|env\.'` to build a complete rotation inventory. Verify Checkmarx KICS Docker image digest against Checkmarx's official advisory for the March 2026 Trivy compromise before any reuse.

Evidence: Before completing eradication, preserve: (1) A full export of the Jenkins credential store metadata (not plaintext secrets) via Jenkins Script Console to document the blast radius — which credential IDs existed and were accessible to pipelines running the malicious plugin. (2) Pipeline execution history showing which jobs ran with the malicious plugin active: `find $JENKINS_HOME/jobs -path '*/builds/*/build.xml' -newer /tmp/may9_marker | xargs grep -l 'checkmarx'` to identify every build that loaded the plugin. (3) Hash of the malicious `.jpi` file (`sha256sum checkmarx.jpi`) for IOC sharing with Checkmarx PSIRT and threat intel feeds. (4) Complete list of environment variables available to affected pipeline builds, reconstructed from Jenkinsfile `environment {}` blocks and `withCredentials` wrappers in build logs — this defines the full set of secrets at risk.

Step 4: Recovery — After credential rotation, verify no unauthorized access occurred against downstream systems using the rotated secrets (check cloud provider access logs, container registry audit logs, source control audit logs). Re-enable affected build pipelines only after confirming plugin removal and full secret rotation. Implement integrity verification (checksum or signature validation) for all reinstalled plugins before restoring production pipeline activity. Monitor pipeline behavior for anomalous outbound connections or unexpected process execution for at least 30 days post-remediation.

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-6 (Audit Record Review, Analysis, and Reporting), NIST CA-7 (Continuous Monitoring), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.4 (Perform Automated Application Patch Management)

Compensating: Pre-reinstatement integrity check: download the replacement Checkmarx Jenkins plugin only from the verified Jenkins Update Center and validate its SHA-256 hash against the value published in Checkmarx's signed advisory before installing. Script this as: `curl -O https://updates.jenkins.io/download/plugins/checkmarx//checkmarx.hpi && sha256sum checkmarx.hpi` — compare output against Checkmarx's published hash. For 30-day post-recovery monitoring without EDR, deploy a Sysmon configuration on all Jenkins build agents targeting process creation (Event ID 1) and network connections (Event ID 3) where the parent image path matches the Jenkins JVM, and forward Sysmon logs to a centralized syslog server using `winlogbeat` (Windows) or `auditbeat` (Linux). Write a Sigma rule targeting outbound connections from `jenkins`

process user to non-RFC1918 addresses during pipeline execution hours as a hunt hypothesis for persistent access. For cloud provider log review, use AWS CloudTrail `LookupEvents` or GCP `gcloud logging read` filtered to the IAM service accounts whose keys were rotated, scoped to the May 9 — rotation date window.

Evidence: Recovery validation requires confirming absence of exploitation artifacts in downstream systems: (1) AWS CloudTrail: query for `AssumeRole`, `GetSecretValue`, `CreateAccessKey`, or `ConsoleLogin` events tied to rotated IAM key IDs between May 9, 2026 and key rotation timestamp. (2) GitHub/GitLab audit logs: review for commits, branch protections changes, or webhook modifications made using the rotated personal access tokens or deploy keys during the exposure window. (3) Container registry (ECR, GCR, Docker Hub): audit image push events from the affected build agent service accounts — TeamPCP may have poisoned container images as a secondary persistence mechanism. (4) Verify Jenkins plugin directory hash consistency post-reinstallation: `find $JENKINS_HOME/plugins -name '*.jpi' | xargs sha256sum > post_recovery_plugin_manifest.txt` and store as baseline for future drift detection.

Step 5: Post-Incident — This attack succeeded because Checkmarx did not rotate publication credentials after its March 2026 Trivy compromise. Internally, review your own credential rotation policy following any third-party supply chain incident affecting tools in your pipeline. Evaluate whether your software supply chain controls include integrity verification (signed artifacts, hash validation) for third-party plugins installed into CI/CD systems. Consider restricting Jenkins plugin installation to an approved, internally mirrored repository where artifacts are scanned before acceptance. Map this incident to NIST SP 800-161r1 (C-SCRM) controls for third-party software integrity assurance.

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 SR-4 (Provenance), NIST SI-2 (Flaw Remediation), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: For teams without enterprise SCRM tooling: (1) Implement an internal Jenkins Update Center mirror using a lightweight tool such as `jenkins-update-center-mirror` (open source) — configure all Jenkins instances to point to the internal mirror URL in `Manage Jenkins → Plugin Manager → Advanced → Update Site`, and gate new plugin versions behind manual hash verification before publishing to the mirror. (2) Write a YARA rule targeting the malicious plugin's known behavioral indicators (credential store API calls followed by outbound HTTP POST) and run it against any new `.jpi` or `.hpi` files before mirror acceptance. (3) Add a Jenkinsfile linting step using the Jenkins Declarative Linter and a custom shared library check that rejects any plugin version not present in an approved manifest file stored in a protected internal repo. (4) Create a post-incident lessons-learned document specifically addressing the gap this incident exposed: no integrity verification trigger fired when TeamPCP published v2026.5.09 using legitimate Checkmarx credentials, meaning signature-only controls are insufficient — hash pinning of approved versions is required.

Evidence: Post-incident documentation must capture: (1) A complete timeline from March 2026 Trivy compromise through May 9, 2026 TeamPCP publication of the malicious Jenkins plugin, including the specific Checkmarx publication credential that was reused — this timeline supports C-SCRM gap analysis against NIST SP 800-161r1. (2) The full inventory of Jenkins jobs, pipelines, and downstream systems that executed with the malicious plugin active, with data classification for each — needed to assess breach notification obligations if PII/PHI-processing pipelines were affected. (3) Evidence of whether any anomalous downstream access occurred (from Step 4 cloud/registry log review) to support a definitive impact determination. (4) A gap analysis comparing the organization's current third-party software intake controls against NIST SI-7 (Software, Firmware, and Information Integrity) requirements, specifically documenting whether hash pinning or signature verification was required and missing for Jenkins plugin installations.

Detection Guidance

Primary detection focus is Jenkins plugin installation history and pipeline execution behavior during the May 9, 2026 exposure window. Query Jenkins plugin management logs or the plugins directory on each Jenkins controller for the presence of checkmarx-ast-scanner v2026.5.09 (or equivalent artifact filename). In SIEM or

EDR, search for anomalous outbound network connections from Jenkins build agent hosts to external IPs, particularly during or immediately following plugin installation events. Look for unexpected child process creation from Jenkins agent processes (e.g., java.exe or jenkins-agent spawning shell interpreters or curl/wget equivalents). Review Jenkins credential store access logs and pipeline execution logs for credential reads not associated with expected job runs. In cloud provider logs (AWS CloudTrail, Azure Monitor, GCP Audit Logs), look for API calls using pipeline-associated service account credentials from unusual source IPs or at unusual times. If you use a secrets manager (Vault, AWS Secrets Manager), check access logs for any pipeline-identity reads during the exposure window. IOCs specific to this incident have not been publicly confirmed in available sources as of this generation; treat any anomalous egress from Jenkins nodes during the exposure window as high-priority for investigation.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`https://plugins.jenkins.io/checkmarx-ast-scanner (v2026.5.09 specifically)`	Malicious plugin version published to official Jenkins Marketplace on May 9, 2026; do not download or retain this version	HIGH

Framework Mappings

MITRE-ATTACK

- **T1078** — Valid Accounts
- **T1552.001** — Credentials In Files
- **T1199** — Trusted Relationship
- **T1059** — Command and Scripting Interpreter
- **T1574** — Hijack Execution Flow
- **T1608.001** — Upload Malware
- **T1588.003** — Code Signing Certificates
- **T1176** — Software Extensions
- **T1195.001** — Compromise Software Dependencies and Development Tools

NIST-800-53R5

- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-7** — Software, Firmware, and Information Integrity

- **CM-3** — Configuration Change Control
- **SR-2** — Supply Chain Risk Management Plan

OWASP-TOP10-2021

- **A08:2021** — Software and Data Integrity Failures
- **A07:2021** — Identification and Authentication Failures
- **A04:2021** — Insecure Design

CIS-V8

- **2.5** — Allowlist Authorized Software
- **2.6** — Allowlist Authorized Libraries
- **16.10** — Apply Secure Design Principles in Application Architectures
- **5.2** — Use Unique Passwords
- **6.3** — Require MFA for Externally-Exposed Applications
- **15.1** — Establish and Maintain an Inventory of Service Providers

ISO-27001-2022

- **A.8.28** — Secure coding
- **A.5.21** — Managing information security in the ICT supply chain

HIPAA-SECURITY

- **164.308(a)(5)(ii)(D)** — Password Management
- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures
- **CC9.2** — Manages risks associated with vendors and business partners

NIST-CSF-2

- **GV.SC-01** — Cybersecurity supply chain risk management program

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1078	Valid Accounts	Defense-Evasion
T1552.001	Credentials In Files	Credential-Access
T1199	Trusted Relationship	Initial-Access
T1059	Command and Scripting Interpreter	Execution
T1574	Hijack Execution Flow	Persistence
T1608.001	Upload Malware	Resource-Development

Technique ID	Technique Name	Tactic
T1588.003	Code Signing Certificates	Resource-Development
T1176	Software Extensions	Persistence
T1195.001	Compromise Software Dependencies and Development Tools	Initial-Access

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/official-checkmarx-j...	T3
Supply Chain Security Incident Update: May 9 - Checkmarx	https://checkmarx.com/blog/supply-chain-security-incident-update-ma...	T3
Update: Ongoing Checkmarx Supply Chain Security Incident	https://checkmarx.com/blog/ongoing-security-updates/	T3
Checkmarx Jenkins AST Plugin Compromised in Supply Chain Attack	https://www.securityweek.com/checkmarx-jenkins-ast-plugin-compromis...	T3
Checkmarx Jenkins Plugin Compromised in Third Supply-Chain ...	https://www.techechelon.com/post/checkmarx-jenkins-plugin-compromis...	T3

Generated 2026-05-12 19:06 UTC by TJS Security Command Center