INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-17 18:30 UTC

CoinbaseCartel Extorts Grafana After Stealing Source Code via Compromised GitHub Token

DATA BREACH | HIGH | CVSS 7.5

SCC Item ID	SCC-DBR-2026-0129
Type	Data Breach
Severity	HIGH
CVSS Base Score	7.5
Affected Products	Grafana (Grafana Cloud, open-source Grafana platform); GitHub Actions CI/CD pipeline
Published	2026-05-17T03:13:33
Discovery Source	Rss

Executive Summary

A threat actor identified as CoinbaseCartel compromised a GitHub access token linked to Grafana's CI/CD pipeline and exfiltrated Grafana's source code repository. Grafana refused the ransom demand and disclosed the incident publicly; no customer data exfiltration has been confirmed. The primary business risk is downstream supply chain exposure: adversaries holding Grafana's source code may identify undisclosed vulnerabilities or attempt to tamper with future build artifacts, affecting any organization that depends on Grafana or Grafana Cloud for observability.

Technical Analysis

The attacker obtained a GitHub personal access token exposed within a Grafana GitHub Actions workflow, consistent with CWE-522 (Insufficiently Protected Credentials) and CWE-312 (Cleartext Storage of Sensitive Information). This token provided sufficient repository-read permissions to clone the Grafana source code, mapping to CWE-284 (Improper Access Control) at the repository authorization layer. No CVE has been assigned; this is a secrets management and workflow misconfiguration incident, not a software vulnerability in Grafana's product itself. MITRE ATT&CK techniques observed include T1552.001 (Credentials in Files), T1528 (Steal Application Access Token), T1213 (Data from Information Repositories), T1530 (Data from Cloud Storage), T1078 (Valid Accounts, applied to the compromised token used as legitimate authentication), T1195.002 (Compromise Software Supply Chain), T1567 (Exfiltration Over Web Service), T1657 (Financial Threats/Extortion), T1537 (Transfer Data to Cloud Account, potential exfiltration route), and T1486 (Data Encrypted for Impact, ransom context). Attribution to CoinbaseCartel, assessed as affiliated with ShinyHunters, Scattered Spider, and LAPSUS$, is medium confidence based on Grafana's public post-incident review and corroboration from StepSecurity. No patch version applies; remediation is workflow hardening, secrets rotation,

and GitHub Actions permission scoping. Grafana's post-incident review is published at grafana.com/blog.

Action Checklist

1. Step 1: Containment, Audit all GitHub Actions workflows in your organization for hardcoded or unnecessarily scoped tokens. Immediately revoke any personal access tokens granted repository-read or write scope that are not actively required. Review GitHub Actions secrets configured at repository and organization level for exposure in workflow logs or environment variables.
2. Step 2: Detection, Query GitHub audit logs (via the GitHub Audit Log API or your SIEM if ingesting GitHub telemetry) for anomalous token authentication events, unexpected repository clone or archive-download activity, and workflow runs initiated by unfamiliar actors or at unusual times. Review Actions workflow run logs for evidence of secrets being echoed or printed. If you depend on Grafana as a build dependency, verify artifact integrity against published checksums.
3. Step 3: Eradication, Replace any exposed or potentially exposed GitHub tokens with fine-grained personal access tokens scoped to the minimum required permissions. Migrate from long-lived tokens to short-lived OIDC-based authentication in GitHub Actions where supported. Implement secret scanning (GitHub Advanced Security or equivalent) to prevent future secrets commits. Reference StepSecurity's hardening guidance published in conjunction with this incident.
4. Step 4: Recovery, After rotating tokens, validate that no unauthorized forks or clones of your repositories were created during the exposure window. Confirm GitHub Actions workflows complete successfully with new credential bindings. Enable branch protection and require signed commits if not already enforced. Monitor Grafana's security advisories page (grafana.com/security/security-advisories/) for any follow-on disclosures of vulnerabilities discovered via the stolen source code.
5. Step 5: Post-Incident, This incident exposes three control gaps common across development environments: (1) absence of least-privilege enforcement on CI/CD tokens, (2) lack of automated secret scanning in pre-commit or CI stages, and (3) no alerting on abnormal repository access volume. Implement mandatory secret scanning, adopt ephemeral OIDC credentials for pipeline authentication, and establish a baseline for expected repository access patterns to enable anomaly detection going forward.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to legal and executive leadership if GitHub Audit Logs confirm that the compromised token had access to repositories containing customer PII, API keys for production systems, or cryptographic signing keys used to sign Grafana release artifacts — any of which would trigger breach notification obligations under GDPR, CCPA, or applicable state law, or could enable downstream supply chain compromise of Grafana-dependent production environments.

Recovery Notes	After token rotation and workflow re-validation, maintain enhanced monitoring of GitHub Audit Logs for `git.clone`, `git.archive`, and `repo.fork` events for a minimum of 90 days, as CoinbaseCartel may retain exfiltrated source code and conduct follow-on reconnaissance to identify exploitable vulnerabilities for future campaigns. Monitor Grafana's security advisory feed continuously during this window, as source code analysis by the threat actor may surface previously undisclosed vulnerabilities in Grafana that are zero-days to the public but known to CoinbaseCartel. Verify all Grafana artifacts consumed in your build pipeline against published SHA256 checksums at each build until Grafana publicly confirms the integrity of their release pipeline post-incident.
Forensic Artifacts	GitHub Audit Log API records for `git.clone`, `git.archive`, `repo.download`, and `oauth_access.create` events — these are the primary artifacts of CoinbaseCartel's bulk source code exfiltration via the compromised CI/CD token, and contain actor IP, timestamp, and repository scope fields critical for scoping the breach. | GitHub Actions workflow run logs (`.github/workflows/` run history) — specific to this attack vector, these logs may contain the compromised token value echoed in plaintext if any workflow step printed environment variables, and will show the exact workflow runs executed under the attacker-controlled context. | GitHub organization secrets inventory snapshot (captured via API before rotation) — documents which secrets were accessible to workflows during the CoinbaseCartel access window, establishing the full scope of potentially exposed credentials beyond the primary compromised token. | Repository fork and clone records from `GET /orgs/{org}/audit-log?phrase=action:repo.fork` — identifies any unauthorized forks created by CoinbaseCartel or associated actors during the incident window, which could indicate ongoing access to exfiltrated code or attempts to establish persistent access. | Grafana release artifact SHA256 checksums from `https://grafana.com/grafana/download` compared against artifacts in your local build cache or artifact repository — detects any tampering with Grafana build outputs that could indicate CoinbaseCartel leveraged stolen source code to introduce backdoors into the release pipeline prior to Grafana's public disclosure.

Per-Action IR Details

Step 1: Containment — Audit all GitHub Actions workflows in your organization for hardcoded or unnecessarily scoped tokens. Immediately revoke any personal access tokens granted repository-read or write scope that are not actively required. Review GitHub Actions secrets configured at repository and organization level for exposure in workflow logs or environment variables.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST AC-2 (Account Management), NIST AC-6 (Least Privilege), CIS 5.1 (Establish and Maintain an Inventory of Accounts), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts)

Compensating: Run `gh api orgs/{ORG}/actions/secrets` and `gh api repos/{ORG}/{REPO}/actions/secrets` via the GitHub CLI to enumerate all configured secrets. Cross-reference against active workflow YAML files using `grep -r 'secrets\.' .github/workflows/` to identify which secrets are actually consumed. For PAT enumeration, use `gh auth status` and query `https://api.github.com/users/{user}/repos?type=all` with each token to map scope. Revoke unused tokens immediately via GitHub Settings > Developer Settings > Personal Access Tokens.

Evidence: Before revoking any tokens, capture a full export of the GitHub organization's active PATs and their last-used timestamps via `GET /orgs/{org}/personal-access-tokens` (requires org:admin scope). Screenshot or export GitHub Actions secrets list at both repo and org level before rotation — this establishes what was exposed during the CoinbaseCartel access window. Preserve the raw `.github/workflows/*.yml` files as they existed at time of incident to document which workflows had access to which secrets.

Step 2: Detection — Query GitHub audit logs (via the GitHub Audit Log API or your SIEM if ingesting GitHub telemetry) for anomalous token authentication events, unexpected repository clone or archive-download activity, and workflow runs initiated by unfamiliar actors or at unusual times. Review Actions workflow run logs for evidence of secrets being echoed or printed. If you depend on Grafana as a build dependency, verify artifact integrity against published checksums.

NIST Phase: Detection Analysis

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

Controls: NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), NIST SI-4 (System Monitoring), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Query GitHub Audit Log API directly without a SIEM: `gh api 'orgs/{ORG}/audit-log?phrase=action:git.clone&per_page=100' --paginate > clone_events.json`. Filter for `git.archive` and `repo.download` action types which indicate bulk source code exfiltration matching the CoinbaseCartel TTP. For Grafana dependency integrity, download the published Grafana release artifact and verify SHA256: `sha256sum -c grafana__checksums.txt`. Use `jq` to parse audit log JSON and isolate token authentication events: `cat audit_log.json | jq '.[] | select(.action=="oauth_access.create" or .action=="personal_access_token.access")'`.

Evidence: Pull GitHub Audit Log entries specifically for `git.clone`, `git.archive`, `repo.download`, and `workflows.completed` events scoped to the 30-day window prior to incident disclosure — CoinbaseCartel's access window maps to this range. Capture `actor_ip`, `actor`, `created_at`, and `repo` fields for each event to identify the source IP and actor identity used during exfiltration. For Grafana-dependent build pipelines, retrieve the Actions workflow run logs from `.github/workflows/` run history and search for any `echo ${{ secrets.* }}` patterns or base64-encoded secret values printed to stdout, which would indicate the token value itself was exposed in run logs.

Step 3: Eradication — Replace any exposed or potentially exposed GitHub tokens with fine-grained personal access tokens scoped to the minimum required permissions. Migrate from long-lived tokens to short-lived OIDC-based authentication in GitHub Actions where supported. Implement secret scanning (GitHub Advanced Security or equivalent) to prevent future secrets commits. Reference StepSecurity's hardening guidance published in conjunction with this incident.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST IA-5 (Authenticator Management), NIST AC-6 (Least Privilege), NIST CM-6 (Configuration Settings), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: For teams without GitHub Advanced Security, deploy `truffleHog` (open source) as a pre-commit hook: `trufflehog git file://. --since-commit HEAD~10 --only-verified` to detect any committed tokens matching GitHub PAT patterns. Add `gitleaks` as a CI check: `gitleaks detect --source . --report-format json --report-path gitleaks-report.json`. For OIDC migration in GitHub Actions without enterprise licensing, configure the `permissions: id-token: write` block in workflow YAML and use `aws-actions/configure-aws-credentials@v4` or equivalent provider action to exchange the OIDC token for short-lived cloud credentials — eliminating the long-lived PAT attack surface that CoinbaseCartel exploited.

Evidence: Before token rotation, use `gh api -H 'Accept: application/vnd.github+json' /repos/{ORG}/{REPO}/git/refs` to enumerate all branches and capture the current HEAD commit SHAs — this creates a pre-rotation baseline to detect any unauthorized commits or branch modifications made during the CoinbaseCartel access window. Run `git log --all --full-history --author-date-order` against a fresh clone and export to file for forensic preservation. Verify no new OAuth Apps or GitHub Apps were registered in the organization during the incident window via `GET /orgs/{org}/installations`.

Step 4: Recovery — After rotating tokens, validate that no unauthorized forks or clones of your repositories were created during the exposure window. Confirm GitHub Actions workflows complete successfully with new credential bindings. Enable branch protection and require signed commits if not already enforced. Monitor Grafana's security advisories page (grafana.com/security/security-advisories/) for any follow-on disclosures of vulnerabilities discovered via the stolen source code.

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 CM-3 (Configuration Change Control), NIST AU-9 (Protection of Audit Information), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 7.3 (Perform Automated Operating System Patch Management)

Compensating: Enumerate unauthorized forks via `gh api 'repos/{ORG}/{REPO}/forks' --paginate | jq '.[].full_name'` and cross-reference against your known authorized fork list. For commit signing verification without enterprise tooling, configure `git config --global commit.gpgsign true` and audit recent commits with `git log --show-signature -10` to identify any unsigned commits introduced during the exposure window. Set up a cron job using `curl` to poll Grafana's security advisories RSS feed (`https://grafana.com/security/security-advisories/index.xml`) daily and diff against a local cache file, alerting on new entries that may disclose vulnerabilities derived from the stolen source code.

Evidence: Query `GET /orgs/{org}/audit-log?phrase=action:repo.fork` for the incident window to identify any forks created using the compromised CoinbaseCartel-accessed token. Capture the full Actions workflow run history post-rotation — specifically the first successful run with new OIDC or rotated PAT credentials — and preserve as the verified clean baseline. For Grafana-dependent supply chains, download and SHA256-verify all Grafana release artifacts consumed by your build pipeline against checksums published at `https://grafana.com/grafana/download` to rule out tampered artifacts before restoring production pipelines.

Step 5: Post-Incident — This incident exposes three control gaps common across development environments: (1) absence of least-privilege enforcement on CI/CD tokens, (2) lack of automated secret scanning in pre-commit or CI stages, and (3) no alerting on abnormal repository access volume. Implement mandatory secret scanning, adopt ephemeral OIDC credentials for pipeline authentication, and establish a baseline for expected repository access patterns to enable anomaly detection going forward.

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-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: Baseline normal repository access volume by exporting 90 days of GitHub Audit Log `git.clone` and `git.fetch` events via the API and computing daily mean/stddev using a simple Python script with `pandas` — flag any day exceeding mean + 3 stddev as an anomaly trigger. Deploy the open-source `harden-runner` GitHub Action from StepSecurity (referenced in the incident disclosure) at the workflow level to detect and block egress of secrets at runtime. For secret scanning without GitHub Advanced Security, configure `pre-commit` with the `detect-secrets` hook (`pip install detect-secrets`) enforced in CI via a required status check, blocking merges that introduce high-entropy strings matching GitHub PAT patterns (`ghp_`, `github_pat_`, `ghs_`).

Evidence: Preserve the complete GitHub Audit Log export covering 90 days prior to and 30 days following the incident as the forensic record of the CoinbaseCartel access pattern — this log is the primary evidence source for any regulatory notification or legal proceedings. Document the three identified control gaps with supporting audit log evidence (e.g., the specific workflow run IDs where long-lived tokens were used, the specific token last-used timestamps showing over-scoped access). Retain the pre- and post-rotation workflow YAML diffs and the secrets inventory snapshots captured during Step 1 as supporting documentation for the lessons-learned report.

Detection Guidance

Primary detection surface is GitHub audit log telemetry. Key event types to query: 'repo.download', 'repo.clone' (if instrumented), 'oauth_access.create', 'personal_access_token.access', and workflow run events from unexpected actors or IP ranges. If ingesting GitHub logs into a SIEM, alert on: single token authenticating to multiple repositories in a short window, repository archive or bulk-download events outside normal CI/CD job patterns, and workflow runs referencing secrets that are then written to stdout or environment output. For

Grafana-specific supply chain risk: verify SHA-256 checksums of Grafana binaries and container images against Grafana's published release hashes before deployment. Monitor for newly published CVEs referencing Grafana internals; an adversary with source access may weaponize undisclosed vulnerabilities before public awareness. No specific IOCs (IPs, domains, hashes) have been publicly attributed to this incident at time of writing.

Framework Mappings

MITRE-ATTACK

- **T1195.002** — Compromise Software Supply Chain
- **T1530** — Data from Cloud Storage
- **T1213** — Data from Information Repositories
- **T1537** — Transfer Data to Cloud Account
- **T1078** — Valid Accounts
- **T1552.001** — Credentials In Files
- **T1528** — Steal Application Access Token
- **T1486** — Data Encrypted for Impact
- **T1567** — Exfiltration Over Web Service
- **T1657** — Financial Theft

NIST-800-53R5

- **CM-7** — Least Functionality
- **SA-9** — External System Services
- **SR-3** — Supply Chain Controls and Processes
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-2** — Account Management
- **AC-6** — Least Privilege
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **CP-9** — System Backup
- **CP-10** — System Recovery and Reconstitution
- **AC-3** — Access Enforcement
- **SR-2** — Supply Chain Risk Management Plan

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control
- **A04:2021** — Insecure Design
- **A07:2021** — Identification and Authentication Failures

CIS-V8

- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process

- **5.2** — Use Unique Passwords
- **6.3** — Require MFA for Externally-Exposed Applications
- **15.1** — Establish and Maintain an Inventory of Service Providers

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets
- **CC7.4** — Responds to identified security incidents
- **CC9.2** — Manages risks associated with vendors and business partners

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control
- **164.308(a)(5)(ii)(D)** — Password Management
- **164.312(d)** — Person or Entity Authentication
- **164.308(a)(6)(ii)** — Response and Reporting

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.34** — Privacy and protection of personal information
- **A.5.21** — Managing information security in the ICT supply chain

NIST-CSF-2

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1195.002	Compromise Software Supply Chain	Initial-Access
T1530	Data from Cloud Storage	Collection
T1213	Data from Information Repositories	Collection
T1537	Transfer Data to Cloud Account	Exfiltration
T1078	Valid Accounts	Defense-Evasion
T1552.001	Credentials In Files	Credential-Access
T1528	Steal Application Access Token	Credential-Access
T1486	Data Encrypted for Impact	Impact
T1567	Exfiltration Over Web Service	Exfiltration
T1657	Financial Theft	Impact

Sources

Source	URL	Tier
Security News	https://thehackernews.com/2026/05/grafana-github-token-breach-led-t...	T3
Security - grafana/grafana	https://github.com/grafana/grafana/security	T3
Grafana security update: post-incident review for GitHub ...	https://grafana.com/blog/grafana-security-update-post-incident-revi...	T3
Grafana GitHub Actions Security Incident	https://www.stepsecurity.io/blog/grafana-github-actions-security-in...	T3
Security Advisories	https://grafana.com/security/security-advisories/	T3

Generated 2026-05-17 18:30 UTC by TJS Security Command Center