INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-20 03:55 UTC

CVE-2026-4525: If a Vault auth mount is configured to pass through the "Authorization" header, and the "Authorizati...

CVE VULNERABILITY | HIGH | CVSS 7.5

SCC Item ID	SCC-CVE-2026-0054
Type	CVE Vulnerability
CVE ID	CVE-2026-4525
Severity	HIGH
CVSS Base Score	7.5
EPSS Score	0.0001 (3th percentile)
Affected Products	HashiCorp Vault and Vault Enterprise, fixed in 2.0.0, 1.21.5, 1.20.10, 1.19.16
Published	2026-04-17T04:16:09.997
Discovery Source	Nvd

Executive Summary

CVE-2026-4525 is a high-severity vulnerability in HashiCorp Vault that causes Vault tokens to be forwarded to external authentication plugin backends when specific header pass-through configurations are active. Organizations using Vault to manage secrets, credentials, or privileged access are at risk of token exposure to any system or party with access to those plugin backends. Exploitation could allow an attacker to move laterally through Vault-protected infrastructure or escalate privileges across systems that rely on Vault for authentication.

Technical Analysis

CVE-2026-4525 (CWE-201: Sensitive Data Exposure) affects HashiCorp Vault and Vault Enterprise. When an auth mount is configured to pass through the 'Authorization' header, and that same header carries the Vault token used to authenticate to Vault itself, Vault incorrectly forwards the token to the downstream auth plugin backend. This breaks the trust boundary: the Vault token reaches an external plugin endpoint that should never receive it. CVSS base score: 7.5 (High). EPSS: 0.00015 (0.015%, low active exploitation probability at time of publication). Affected versions: all releases prior to 2.0.0, 1.21.5, 1.20.10, and 1.19.16. Fixed in: 2.0.0, 1.21.5, 1.20.10, 1.19.16. MITRE techniques: T1552.001 (Credentials in Files, token capture from transit), T1550.001 (Application Access Token, use of captured token for access). No known active exploitation; not listed in CISA

KEV as of configuration date.

Action Checklist

1. Step 1: Containment, Identify all Vault auth mounts configured with 'Authorization' header pass-through enabled. Disable or restrict access to those mounts until patching is complete. If the plugin backend is internet-accessible or shared with untrusted parties, treat any Vault tokens that transited those mounts as compromised and revoke them immediately via 'vault token revoke' or lease revocation.
2. Step 2: Detection, Review Vault audit logs for auth requests where the Authorization header was forwarded to a plugin backend. Query audit log entries for auth mount paths with header pass-through enabled and correlate with downstream plugin backend access logs for unexpected token usage. Look for token activity originating from plugin backend IP addresses rather than expected client IPs. Check for T1550.001-style token reuse from anomalous sources.
3. Step 3: Eradication, Upgrade Vault or Vault Enterprise to a fixed version: 2.0.0, 1.21.5 (for 1.21.x), 1.20.10 (for 1.20.x), or 1.19.16 (for 1.19.x). Follow HashiCorp's official upgrade documentation for your deployment type. After upgrading, audit all auth mount configurations and remove 'Authorization' header pass-through where it is not operationally required.
4. Step 4: Recovery, After patching, rotate all Vault tokens that were active on affected auth mounts during the vulnerable period. Verify the upgrade by confirming the running Vault version via 'vault status'. Re-enable affected auth mounts and test that the Authorization header is no longer forwarded to plugin backends. Monitor Vault audit logs for 48-72 hours post-remediation for anomalous token usage patterns.
5. Step 5: Post-Incident, Conduct a configuration review of all Vault auth mounts to enforce least-privilege header forwarding. Implement network segmentation between Vault and plugin backend endpoints to limit exposure if the flaw recurs. Add Vault token forwarding behavior to your Vault hardening checklist. Review your secrets management policy to ensure Vault tokens are scoped to minimum required permissions, limiting blast radius if a token is captured.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate to CISO and legal/compliance immediately if Vault audit logs confirm that tokens with access to PII, PHI, financial data, or cloud provider credentials (e.g., AWS IAM, Azure AD) were forwarded to plugin backends — particularly if those backends are internet-accessible or operated by third parties — as this may constitute a reportable data breach under GDPR, HIPAA, or PCI-DSS breach notification obligations.
Recovery Notes	After revoking exposed tokens and upgrading to a fixed Vault version (2.0.0, 1.21.5, 1.20.10, or 1.19.16), all secrets, credentials, and dynamic leases accessible via the compromised tokens must be rotated in addition to the tokens themselves — the token is the access mechanism, but the secrets it could read are the actual blast radius. Monitor Vault audit logs continuously for 72 hours post-remediation using the jq-based query developed in Step 2 to detect any replayed pre-rotation tokens (T1550.001), and validate that plugin backend HTTP access logs no longer show inbound Authorization headers originating from Vault's egress IP. Consider placing Vault in a temporary read-only or restricted mode during the recovery window if token rotation across all dependent services cannot be completed atomically.

Forensic Artifacts	Vault JSON audit log (/var/log/vault/audit.log or syslog forward target): Contains 'request.headers' fields that directly evidence whether the Authorization header was forwarded on plugin-backed auth mount requests during the vulnerable period — the primary artifact for scope determination. | Plugin backend HTTP server access logs (Apache/nginx/custom): Inbound requests from Vault's egress IP carrying 'Authorization: Bearer ' headers represent the forwarded Vault token as received by the backend — captures the actual token value exposed to the third-party system. | Vault token accessor list snapshot (captured via 'vault list auth/token/accessors' before revocation): Maps all active tokens to their issuing auth mount, enabling correlation of which tokens transited affected mounts and were therefore potentially exposed. | Vault auth mount configuration dump (vault auth list -detailed -format=json): Preserves the exact 'passthrough_request_headers' configuration at time of incident, establishing which mounts were misconfigured and for how long — critical for regulatory reporting and blast radius assessment. | Downstream API and service access logs for systems protected by Vault-managed credentials: Any service whose credentials were accessible via tokens that transited affected mounts must be reviewed for access events originating from plugin backend IP addresses, evidencing lateral movement via captured tokens (MITRE T1550.001).

Per-Action IR Details

Step 1: Containment — Identify all Vault auth mounts configured with 'Authorization' header pass-through enabled. Disable or restrict access to those mounts until patching is complete. If the plugin backend is internet-accessible or shared with untrusted parties, treat any Vault tokens that transited those mounts as compromised and revoke them immediately via 'vault token revoke' or lease revocation.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST AC-2 (Account Management), CIS 5.3 (Disable Dormant Accounts), CIS 6.2 (Establish an Access Revoking Process)

Compensating: Run 'vault auth list -detailed' and parse the output for 'passthrough_request_headers' values that include 'Authorization'; pipe through 'grep -i authorization' to quickly surface affected mounts. For bulk token revocation without enterprise tooling, use 'vault token revoke -accessor ' per token or 'vault lease revoke -prefix auth//' to sweep all leases under a given mount in one command.

Evidence: Before disabling mounts, capture a full snapshot of current auth mount configurations via 'vault auth list -detailed -format=json > vault_auth_mounts_snapshot.json'. Preserve Vault audit log files from the period the mount was active — typically located at the path configured in 'vault audit list', commonly '/var/log/vault/audit.log' or a syslog forward target. Record the accessor IDs of all active tokens on affected mounts via 'vault list auth/token/accessors' before revocation, as these are needed to reconstruct the scope of potential token exposure.

Step 2: Detection — Review Vault audit logs for auth requests where the Authorization header was forwarded to a plugin backend. Query audit log entries for auth mount paths with header pass-through enabled and correlate with downstream plugin backend access logs for unexpected token usage. Look for token activity originating from plugin backend IP addresses rather than expected client IPs. Check for T1550.001-style token reuse from anomalous sources.

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-3 (Content of Audit Records), NIST SI-4 (System Monitoring), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Use 'jq' to parse Vault's JSON audit log: 'jq '. | select(.request.mount_type == "plugin" and (.request.headers.Authorization // empty))' /var/log/vault/audit.log' to isolate requests where the Authorization header was present on plugin-backed auth mounts. Cross-reference plugin backend access logs (e.g., Apache/nginx access

logs for HTTP-based plugin backends) for requests arriving from Vault's egress IP that carry a Bearer token in the Authorization header — these represent the forwarded Vault token. For MITRE T1550.001 (Use Alternate Authentication Material: Application Access Token) detection, use a Sigma rule matching Vault audit log entries where 'auth.client_token' appears in requests sourced from the plugin backend's IP range rather than known client CIDRs.

Evidence: Vault audit log entries (JSON format) showing 'request.headers' containing an 'Authorization' key on auth mount paths with pass-through enabled — these entries directly evidence token forwarding. Plugin backend HTTP server access logs showing inbound requests from Vault's IP with an 'Authorization: Bearer ' header — this is the forwarded Vault token as seen by the backend. Downstream Vault API audit log entries where a token originally issued via the affected auth mount is subsequently used from an IP address attributable to the plugin backend or an unexpected external host, evidencing potential token reuse (T1550.001).

Step 3: Eradication — Upgrade Vault or Vault Enterprise to a fixed version: 2.0.0, 1.21.5 (for 1.21.x), 1.20.10 (for 1.20.x), or 1.19.16 (for 1.19.x). Follow HashiCorp's official upgrade documentation for your deployment type. After upgrading, audit all auth mount configurations and remove 'Authorization' header pass-through where it is not operationally required.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST CM-6 (Configuration Settings), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: For teams running Vault via systemd, use 'vault version' pre- and post-upgrade to confirm the binary change; pin the check in a shell script: 'vault version | grep -E "1\.21\.5|1\.20\.10|1\.19\.16|2\.0\.0" || echo UPGRADE_REQUIRED'. After upgrading, audit all mounts for residual header pass-through config using 'vault auth list -detailed -format=json | jq '.[] | select(.config.passthrough_request_headers[]? | ascii_downcase == "authorization") | .path'' — any output requires manual remediation via 'vault auth tune -passthrough-request-headers="" ' to clear the offending configuration.

Evidence: Before upgrading, capture 'vault status -format=json > vault_status_pre_upgrade.json' and store the binary hash ('sha256sum /usr/local/bin/vault > vault_binary_hash_pre.txt') to establish a pre-patch baseline. Preserve the full auth mount configuration dump ('vault auth list -detailed -format=json > vault_mounts_pre_eradication.json') as forensic evidence of which mounts had header pass-through active. After upgrade, re-run both commands and retain post-upgrade artifacts to document the eradication action for audit and post-incident review.

Step 4: Recovery — After patching, rotate all Vault tokens that were active on affected auth mounts during the vulnerable period. Verify the upgrade by confirming the running Vault version via 'vault status'. Re-enable affected auth mounts and test that the Authorization header is no longer forwarded to plugin backends. Monitor Vault audit logs for 48-72 hours post-remediation for anomalous token usage patterns.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST AC-2 (Account Management), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST SI-2 (Flaw Remediation), CIS 5.1 (Establish and Maintain an Inventory of Accounts)

Compensating: For token rotation without automated tooling, iterate through the accessor list captured during containment and revoke each: 'for acc in $(cat accessors.txt); do vault token revoke -accessor $acc; done'. Validate that the Authorization header is no longer forwarded post-upgrade by issuing a test auth request against the previously affected mount and inspecting plugin backend logs for the presence or absence of an Authorization header — absence confirms eradication. For the 48-72 hour monitoring window, set up a cron job that runs the jq query from Step 2 against the rolling audit log every 15 minutes and writes matches to an alert file: 'jq '. | select(.request.headers.Authorization // empty)' /var/log/vault/audit.log >> /var/log/vault/token_forward_alerts.log'.

Evidence: Vault audit log entries post-upgrade confirming auth requests on formerly affected mounts no longer contain 'request.headers.Authorization' — this is the primary indicator of successful eradication. Token accessor inventory pre- and post-rotation to confirm all potentially exposed tokens have been invalidated. Plugin backend HTTP access logs for the 48-72 hour monitoring window, specifically checking for any inbound requests carrying a Vault-issued Bearer

token that predates the rotation, which would indicate a previously captured token being replayed (T1550.001).

Step 5: Post-Incident — Conduct a configuration review of all Vault auth mounts to enforce least-privilege header forwarding. Implement network segmentation between Vault and plugin backend endpoints to limit exposure if the flaw recurs. Add Vault token forwarding behavior to your Vault hardening checklist. Review your secrets management policy to ensure Vault tokens are scoped to minimum required permissions, limiting blast radius if a token is captured.

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 SC-7 (Boundary Protection), NIST AC-6 (Least Privilege), NIST CM-6 (Configuration Settings), CIS 4.2 (Establish and Maintain a Secure Configuration Process for Network Infrastructure), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Codify the auth mount header review as a recurring osquery scheduled query against the Vault configuration database, or as a shell script run on a weekly cron: 'vault auth list -detailed -format=json | jq '[.[] | {path: .path, passthrough_headers: .config.passthrough_request_headers}]' > vault_header_audit_$(date +%F).json'. For network segmentation without enterprise firewalling, implement host-based firewall rules (iptables/nftables) on the Vault host to restrict outbound connections to plugin backend IPs to only the specific ports required, blocking all other egress from the Vault process UID. Add a Vault policy audit step to the Vault hardening checklist that validates all token policies include explicit 'max_ttl' and capability scoping, limiting the window and blast radius of any future token capture.

Evidence: Completed configuration audit report ('vault auth list -detailed -format=json' output, dated post-incident) documenting that no auth mounts retain 'Authorization' in 'passthrough_request_headers' — serves as the formal closure artifact. Network flow logs or firewall rule exports documenting the new segmentation controls between Vault and plugin backend endpoints. Updated Vault policy files (HCL) for any tokens identified as over-privileged during the incident review, retained as evidence of policy hardening. Lessons-learned document referencing the specific CVE-2026-4525 mechanism (header pass-through misconfiguration leading to token forwarding) to anchor future tabletop exercises.

Detection Guidance

Enable and review Vault audit logs (file or syslog backend). Filter for requests to auth mount paths where 'passthrough_request_headers' includes 'Authorization'. Correlate those entries with downstream plugin backend access logs to identify any requests where the Vault token appeared in the backend's received headers. Behavioral indicator: Vault token activity originating from plugin backend hostnames or IPs rather than expected client sources (T1550.001 pattern). If SIEM is in use, create a detection rule for Vault audit log entries where the client token is present in outbound plugin requests. Note: detection requires Vault audit logging to be enabled prior to the event; if audit logging was not active, retroactive detection is not possible from Vault logs alone.

Framework Mappings

MITRE-ATTACK

- **T1552.001** — Credentials In Files
- **T1550.001** — Application Access Token

CIS-V8

- **6.3** — Require MFA for Externally-Exposed Applications
- **5.4** — Restrict Administrator Privileges to Dedicated Administrator Accounts

HIPAA-SECURITY

- **164.312(d)** — Person or Entity Authentication

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures
- **CC6.3** — Authorizes, modifies, or removes access

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities

NIST-800-53R5

- **AC-6** — Least Privilege

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1552.001	Credentials In Files	Credential-Access
T1550.001	Application Access Token	Defense-Evasion

Sources

Source	URL	Tier
nvd	https://nvd.nist.gov/vuln/detail/CVE-2026-4525	T1
CVE-2026-4525 - Red Hat Customer Portal	https://access.redhat.com/security/cve/cve-2026-4525	T3
CVE-2026-4525 Security Vulnerability Analysis & Exploit Details	https://cve.akaoma.com/cve-2026-4525	T3
CVE-2026-4525 | Mondoo Vulnerability Intelligence	https://mondoo.com/vulnerability-intelligence/vulnerability/CVE-202...	T3
CVE-2026-4525 HashiCorp Vault/Vault Enterprise Header insertion ...	https://vuldb.com/vuln/357996	T3

Generated 2026-04-20 03:55 UTC by TJS Security Command Center