INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-12 06:38 UTC

GHSA-qccp-gfcp-xxvc: urllib3: Sensitive headers forwarded across origins in proxied low-level redirec

CVE VULNERABILITY | MEDIUM | CVSS 6.1

SCC Item ID	SCC-CVE-2026-0157
Type	CVE Vulnerability
CVE ID	CVE-2026-44431
Severity	MEDIUM
CVSS Base Score	6.1
Affected Products	urllib3 (PyPI) >= 1.23, < 2.7.0
Published	2026-05-11T14:51:20Z
Discovery Source	Osv

Executive Summary

A credential leakage vulnerability in urllib3, a widely used Python HTTP library, allows sensitive authentication headers to be forwarded to unintended third-party servers during proxied redirects. Applications using urllib3 versions 1.23 through 2.6.x that route traffic through proxies and follow redirects are exposed. The business risk is unauthorized disclosure of credentials or session tokens to external servers, potentially enabling account takeovers or unauthorized access to internal systems.

Technical Analysis

CVE-2026-44431 (GHSA-qccp-gfcp-xxvc) is a header leakage vulnerability in urllib3's redirect handling logic. When a proxied request follows a redirect to a different origin, urllib3 fails to strip sensitive headers, specifically Authorization and Cookie headers, before forwarding the request to the new destination. This violates the expected cross-origin security boundary. Affected versions: urllib3 >= 1.23, = 2.7.0.

Action Checklist

1. Identification: Audit your Python application inventory for urllib3 versions >= 1.23 and < 2.7.0. Prioritize services that use proxies and follow HTTP redirects, particularly those transmitting Authorization or Cookie headers. Temporarily disable automatic redirect following (set redirect=False) in high-risk urllib3 usage

until patching is complete.

2. Assessment: Search application dependency manifests (requirements.txt, Pipfile.lock, pyproject.toml, conda environment files) and runtime package lists (pip list, pip freeze) for urllib3 < 2.7.0. In proxy and API gateway logs, look for cross-origin redirect chains where the destination host differs from the original request host; these represent the conditions under which leakage occurs.

3. Remediation: Upgrade urllib3 to version 2.7.0 or later via pip (pip install --upgrade urllib3). If direct upgrade is blocked by dependency constraints, audit downstream packages (requests, boto3, kubernetes client) for bundled urllib3 versions and upgrade those packages as well. Validate the installed version post-upgrade with pip show urllib3.

4. Validation: After upgrading, verify that Authorization and Cookie headers are stripped on cross-origin redirects by reviewing application behavior in a test environment. Monitor proxy and access logs for anomalous outbound requests carrying authentication headers to unexpected external hosts. Confirm no credentials were forwarded to unintended origins by reviewing historical proxy logs for the exposure window.

5. Monitoring: This vulnerability exposes a control gap in third-party library version management and dependency monitoring. Review your software composition analysis (SCA) tooling coverage against OSV and GitHub Advisory feeds. Establish or validate a policy requiring that library upgrades for Medium+ severity CVEs in network-layer dependencies complete within your standard patching SLA, aligned with your GRC program's standards.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate to incident response lead and legal/compliance if historical proxy log review confirms outbound requests carried Authorization or Cookie headers to external hosts outside your organization's known domain inventory during the exposure window — this finding triggers breach notification assessment under applicable data protection regulations (GDPR Article 33, US state breach notification laws) if the forwarded credentials are linked to user accounts or PII.
Recovery Notes	After upgrading to urllib3 2.7.0, rotate all API keys, Bearer tokens, and session credentials that were in active use by applications routing traffic through proxies with redirect-following enabled during the exposure window — assume any such credential may have been forwarded to an unintended redirect target. Monitor outbound proxy traffic for at least 30 days post-remediation for anomalous authentication attempts from external hosts that could indicate harvested credentials being replayed. Confirm no downstream packages (requests, boto3, kubernetes-client) re-pin a vulnerable urllib3 version after the next dependency update cycle by running `pip-audit` as part of the post-deployment validation gate.

Forensic Artifacts	Proxy access logs (Squid /var/log/squid/access.log, nginx /var/log/nginx/access.log, HAProxy /var/log/haproxy.log) filtered for HTTP 301/302/307/308 redirect chains where the destination FQDN differs from the originating request host — these sequences are the direct trigger condition for CVE-2026-44431 header forwarding | pip freeze output and lockfile snapshots (requirements.txt, Pipfile.lock, pyproject.toml) from all affected application hosts timestamped before and after remediation, establishing the vulnerable version window and dependency chain (requests, boto3, kubernetes-client) that bundled the affected urllib3 build | Application environment variable snapshots capturing HTTP_PROXY, HTTPS_PROXY, and ALL_PROXY values — these identify which proxy endpoints received traffic and are the servers to which Authorization and Cookie headers would have been forwarded during cross-origin redirects | Network packet captures (via tcpdump or Wireshark) of application outbound traffic on port 80/443 taken during the test-environment validation step, confirming pre-patch header leakage versus post-patch header stripping on redirect sequences to a different-origin host | pip show urllib3 output (including Requires and Required-by fields) from each affected host, preserved pre-patch, documenting the exact installed version and the full transitive dependency chain that introduced the vulnerable urllib3 instance into each environment

Per-Action IR Details

Containment — Audit your Python application inventory for urllib3 versions >= 1.23 and < 2.7.0. Prioritize services that use proxies and follow HTTP redirects, particularly those transmitting Authorization or Cookie headers. Temporarily disable automatic redirect following (set redirect=False) in high-risk urllib3 usage until patching is complete.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST CM-8 (System Component Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory)

Compensating: Run `pip list --format=json | python3 -c "import sys,json; [print(p['name'],p['version']) for p in json.load(sys.stdin) if 'urllib3' in p['name'].lower()]"` across all application hosts. For containerized workloads, execute `docker exec pip show urllib3` per running container. Immediately grep application source for `redirect=True` or absence of `redirect=False` in urllib3/requests call sites: `grep -rn 'urllib3\|requests.get\|requests.post\|HTTPAdapter' /opt/app --include='*.py' | grep -v 'redirect=False'`.

Evidence: Before disabling redirects, capture the current urllib3 call graph: run `pip show urllib3` to record the exact installed version and dependent packages (Requires, Required-by fields). Snapshot all outbound proxy configurations from environment variables (`HTTP_PROXY`, `HTTPS_PROXY`, `ALL_PROXY`) and from application config files — these define which proxy endpoints could have received leaked Authorization or Cookie headers during cross-origin redirects.

Detection — Search application dependency manifests (requirements.txt, Pipfile.lock, pyproject.toml, conda environment files) and runtime package lists (pip list, pip freeze) for urllib3 < 2.7.0. In proxy and API gateway logs, look for cross-origin redirect chains where the destination host differs from the original request host — these represent the conditions under which leakage occurs.

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: For manifest scanning without an SCA tool, run: `find / -name 'requirements*.txt' -o -name 'Pipfile.lock' -o -name 'pyproject.toml' 2>/dev/null | xargs grep -l 'urllib3'` then extract pinned versions. For proxy log analysis (Squid/nginx), run: `awk '{print $7}' /var/log/squid/access.log | grep -E 'http[s]?://' | sort | uniq -c | sort -rn` to surface

redirect destination hosts, then cross-reference against your known application host allowlist to identify unexpected cross-origin redirect targets where Authorization headers would have been forwarded.

Evidence: Extract proxy access logs (Squid: `/var/log/squid/access.log`; nginx: `/var/log/nginx/access.log`; HAProxy: `/var/log/haproxy.log`) for the full exposure window (urllib3 >= 1.23 install date through patch date). Filter for HTTP 301/302/303/307/308 redirect responses followed immediately by a request to a different FQDN from the same client IP — this is the specific redirect chain pattern that triggers CVE-2026-44431 header forwarding. Also preserve `pip freeze` output and lockfile snapshots from all affected hosts as evidence of the vulnerable version window.

Eradication — Upgrade urllib3 to version 2.7.0 or later via pip (pip install --upgrade urllib3). If direct upgrade is blocked by dependency constraints, audit downstream packages (requests, boto3, kubernetes client) for bundled urllib3 versions and upgrade those packages as well. Validate the installed version post-upgrade with pip show urllib3.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST CM-3 (Configuration Change Control), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: For environments with dependency conflicts, run `pip install --upgrade urllib3 requests boto3` together to resolve transitive pins in one pass, then validate with `pip show urllib3 requests boto3 | grep -E 'Name|Version|Requires'`. For containerized apps, rebuild base images with `urllib3>=2.7.0` pinned in requirements.txt and redeploy — do not patch running containers in place without rebuilding the image layer. Confirm the fix with: `python3 -c "import urllib3; print(urllib3.__version__)"` on each affected host and record the output as remediation evidence.

Evidence: Before upgrading, preserve a forensic copy of the vulnerable package metadata: `pip show urllib3 > /tmp/urllib3_pre_patch_$(hostname)_$(date +%Y%m%d).txt` and capture `pip list --format=freeze > /tmp/pip_freeze_pre_patch_$(hostname)_$(date +%Y%m%d).txt`. These files establish the pre-remediation state for audit trail purposes under NIST AU-11 (Audit Record Retention) and document which specific dependency chain (requests, boto3, kubernetes-client) bundled the vulnerable urllib3 version.

Recovery — After upgrading, verify that Authorization and Cookie headers are stripped on cross-origin redirects by reviewing application behavior in a test environment. Monitor proxy and access logs for anomalous outbound requests carrying authentication headers to unexpected external hosts. Confirm no credentials were forwarded to unintended origins by reviewing historical proxy logs for the exposure window.

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), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Validate the fix behaviorally using a free test harness: set up a local redirect server with Python's `http.server` redirecting to a different-origin listener (e.g., `localhost:8080` → `localhost:9090`) and capture headers with Wireshark or `tcpdump -i lo -A port 9090`; confirm that `Authorization` and `Cookie` headers are absent in the redirected request when using urllib3 2.7.0. For historical proxy log review, use: `grep -E '(Authorization|Cookie)' /var/log/squid/access.log` (if headers are logged) or analyze CONNECT tunnel metadata to identify cross-origin redirect sequences that occurred during the exposure window.

Evidence: Pull proxy logs for the full exposure window and search for outbound requests to hosts outside your organization's known API domains that immediately follow a redirect response — specifically filter for sequences where the originating application is known to transmit `Authorization: Bearer`, `Authorization: Basic`, or `Cookie:` headers. Preserve any such log entries as potential credential exposure evidence; if PII-linked session tokens or OAuth credentials are involved, this evidence will be required for breach assessment under applicable regulatory frameworks.

Post-Incident — This vulnerability exposes a control gap in third-party library version management and dependency monitoring. Review your software composition analysis (SCA) tooling coverage against OSV and

GitHub Advisory feeds. Establish or validate a policy requiring that library upgrades for Medium+ severity CVEs in network-layer dependencies complete within a defined SLA, aligned with your GRC program's patching standards.

NIST Phase: Post Incident

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

Controls: NIST SI-2 (Flaw Remediation), NIST SI-5 (Security Alerts, Advisories, and Directives), NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST RA-3 (Risk Assessment), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: Implement free SCA coverage immediately using `pip-audit` (`pip install pip-audit && pip-audit`) against all application virtual environments — it queries OSV.dev and PyPA advisory feeds and will flag urllib3 CVEs including CVE-2026-44431. Integrate into CI/CD by adding `pip-audit --fail-on-vuln` as a pipeline gate. For ongoing monitoring without paid tooling, subscribe to GitHub Advisory Database notifications for urllib3 and top-10 transitive dependencies (requests, boto3, cryptography, certifi) via GitHub's free dependency alert feature on all application repositories.

Evidence: Document the lessons-learned record per NIST 800-61r3 §4: record the detection gap (how long urllib3 < 2.7.0 was present without alert), the exposure window (first vulnerable install date to patch date), and whether any cross-origin redirect sequences in historical proxy logs confirmed or ruled out actual credential forwarding. This record supports both the GRC patching SLA policy update and any regulatory disclosure assessment if session tokens tied to user accounts were found in the exposure evidence.

Detection Guidance

Identify exposure by running pip freeze or pip list on all Python application hosts and comparing urllib3 versions against the affected range (>= 1.23, < 2.7.0). In containerized environments, scan image layers using your SCA toolchain against GHSA-qccp-gfcp-xxvc. For runtime detection, review proxy egress logs and API gateway access logs for redirect events where the Location header points to a different origin (different scheme, host, or port) on requests that carry Authorization or Cookie headers. Example Splunk query: index=proxy_logs status_code IN (301,302,307,308) | where src_host != dst_host AND request_headers CONTAINS "Authorization" | stats count by src_host, dst_host, location_header. No known public exploit code or IOCs are associated with this vulnerability at this time.

Framework Mappings

MITRE-ATTACK

- **T1557** — Adversary-in-the-Middle
- **T1552.001** — Credentials In Files

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control

NIST-800-53R5

- **AC-3** — Access Enforcement
- **SC-28** — Protection of Information at Rest
- **SR-2** — Supply Chain Risk Management Plan

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control
- **164.312(d)** — Person or Entity Authentication

CIS-V8

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

SOC2-TSC

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

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **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
T1557	Adversary-in-the-Middle	Credential-Access
T1552.001	Credentials In Files	Credential-Access

Sources

Source	URL	Tier
osv	https://osv.dev/vulnerability/GHSA-qccp-gfcp-xxvc	T3
CVE-2026-44431 - Docker Scout	https://scout.docker.com/vulnerabilities/id/CVE-2026-44431?t=pypi&a...;	T3
urllib3: Sensitive headers forwarded across origins in proxied low ...	https://github.com/advisories/GHSA-qccp-gfcp-xxvc	T3
CVE-2026-44431 - Exploits & Severity - Feedly	https://feedly.com/cve/CVE-2026-44431	T3

Source	URL	Tier
CVE-2026-31431: Copy Fail vulnerability enables Linux ... - Microsoft	https://www.microsoft.com/en-us/security/blog/2026/05/01/cve-2026-3...	T1
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-44431	T1

Generated 2026-05-12 06:38 UTC by TJS Security Command Center