INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-22 19:03 UTC

DifyTap: Four Authorization Flaws in Dify Enable Unauthenticated Cross-Tenant AI Chat Exfiltration

CVE VULNERABILITY | HIGH | CVSS 7.5

SCC Item ID	SCC-CVE-2026-0343
Type	CVE Vulnerability
CVE ID	CVE-2024-5846, CVE-2026-41947, CVE-2026-41948, CVE-2026-41949, CVE-2026-41950
Severity	HIGH
CVSS Base Score	7.5
EPSS Score	0.0046 (37th percentile)
Affected Products	Dify open-source agentic AI platform, versions prior to 1.14.2
Published	2026-06-22T12:13:28
Discovery Source	Rss

Executive Summary

Zafran Security disclosed four vulnerabilities in Dify, a widely deployed open-source platform used to build and operate AI agent workflows, that together allow attackers to read AI conversations across organizational tenant boundaries, access internal APIs, and exfiltrate uploaded documents with minimal authentication. Two of the four individual CVEs carry critical CVSS scores (9.1 and 9.4 per vendor disclosure); the aggregate item-level CVSS base is 7.5 (high). Three of the four flaws affect Dify's cloud service with cross-tenant impact. Organizations running Dify in production, particularly in multi-tenant or cloud-hosted configurations, face direct risk of AI conversation data theft and unauthorized document exfiltration until v1.14.2 is applied.

Technical Analysis

Zafran Security's 'DifyTap' research identified four authorization vulnerabilities in Dify (open-source agentic AI platform) affecting versions prior to 1.14.2. CVE IDs CVE-2026-41947 through CVE-2026-41950 are attributed to this disclosure; NVD publication is pending as of this analysis, so attribution carries medium confidence. The four flaws map to CWE-285 (Improper Authorization), CWE-862 (Missing Authorization), CWE-639 (Authorization Bypass Through User-Controlled Key), and CWE-22 (Path Traversal). Collectively they enable: cross-tenant AI conversation read access (T1602, T1530), traversal of the internal Plugin Daemon API (T1083,

T1190), and unauthenticated or minimally authenticated document exfiltration (T1119, T1567). Two CVEs carry critical CVSS scores (9.1 and 9.4 per vendor disclosure); the aggregate item-level CVSS base is reported at 7.5 (high). Three of four flaws are patched in v1.14.2. CVE-2026-41948, a path traversal flaw (CWE-22), remains unpatched as of this analysis; no vendor timeline for a fix has been published. EPSS score is 0.00464 (36th percentile); CISA KEV listing is not confirmed. Note: CVE-2024-5846 (PDFium) appearing in raw input data is unrelated to this disclosure and has been excluded.

Action Checklist

1. Step 1: Containment. Identify all Dify deployments in your environment (cloud-hosted and self-hosted). Restrict network access to Dify's API endpoints and Plugin Daemon interfaces using perimeter firewall rules (layer 3/4) and/or WAF policies (layer 7) as appropriate for your architecture. For cloud-hosted Dify tenants, review access logs immediately for cross-tenant API calls or anomalous document retrieval requests. Disable external API access to Dify where not operationally required until patching is complete.
2. Step 2: Detection. Query application and API gateway logs for requests to Dify's Plugin Daemon endpoints from users or service accounts outside expected tenant scopes. Look for path traversal patterns (e.g., '../' sequences) in API request URLs targeting document or file endpoints. Correlate against MITRE T1083 (file and directory discovery) and T1602 (data from configuration repositories) behavioral indicators. Review audit logs per NIST AU-6 (Audit Record Review) and CIS 8.2 (Collect Audit Logs) for anomalous cross-tenant conversation retrieval.
3. Step 3: Eradication. Upgrade all self-hosted Dify instances to v1.14.2 or later per the vendor release at https://github.com/langgenius/dify. For CVE-2026-41948 (path traversal, unpatched), apply WAF rules to block path traversal sequences in requests targeting Dify document/file endpoints as an interim control until the vendor issues a fix. Rotate API keys and session tokens for all Dify tenants as a precaution per credential rotation procedures (NIST IA-4).
4. Step 4: Recovery. After upgrading to v1.14.2, validate that authorization controls enforce tenant boundary isolation by testing cross-tenant API requests and confirming they are rejected. Review access control lists on document storage associated with Dify (NIST AC-3, CIS 3.3). Monitor Dify API logs for 30 days post-patch for residual anomalous access patterns. Confirm CVE-2026-41948 path traversal is blocked at the WAF layer until the vendor patch is released.
5. Step 5: Post-Incident. Conduct a review of authorization architecture for all AI/ML platforms in your environment, specifically evaluating tenant isolation controls and internal API exposure (NIST AC-4, AC-6). Evaluate whether Plugin Daemon and other internal service APIs are accessible from external network segments and remediate as needed. Document control gaps and update your AI platform security baseline. Consider adding local account monitoring and user account permissions reviews to your AI platform hardening checklist.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to legal, privacy, and executive leadership immediately if Dify API logs show evidence of successful cross-tenant conversation retrieval or document access (HTTP 200 responses on `/v1/messages`, `/v1/files`, or `/v1/datasets` from unexpected tenant IDs), as this constitutes confirmed data exfiltration across organizational boundaries that may trigger breach notification obligations under GDPR, CCPA, or HIPAA depending on the nature of AI conversation content and uploaded documents.
Recovery Notes	After deploying v1.14.2, perform explicit cross-tenant authorization testing against all five CVE-patched endpoints before restoring external API access, and verify that Plugin Daemon (port 5002) is firewalled from all external and non-administrative network segments. Monitor Dify API and WAF logs daily for the first two weeks post-patch, then weekly for 30 days, specifically watching for path traversal sequences and tenant ID anomalies that may indicate an attacker retesting whether the vulnerabilities were fully closed. Retain all pre-patch logs for a minimum of 90 days to support any post-discovery forensic investigation or regulatory inquiry into what data may have been accessed during the exposure window.
Forensic Artifacts	Dify API access logs (nginx or application-layer): Request URIs containing `/v1/messages`, `/v1/files`, `/v1/datasets`, and Plugin Daemon paths with mismatched or unexpected `X-Tenant-ID` header values — the primary forensic signature of CVE-2024-5846 and CVE-2026-41947 cross-tenant authorization bypass exploitation | WAF or reverse proxy logs: HTTP requests containing path traversal sequences (`../`, `%2e%2e%2f`, `%252e%252e`) in URLs targeting Dify document or file endpoints — the specific exploitation fingerprint of CVE-2026-41948, which was unpatched at time of disclosure | Redis session store dump (`redis-cli KEYS 'session:*'` with HGETALL): Active or recently expired session tokens associated with service accounts or API keys that made cross-tenant requests, preserving attacker session identifiers before credential rotation destroys them | Dify PostgreSQL database query history and `conversations`/`messages`/`uploaded_files` table access timestamps: Row-level access timestamps on records belonging to victim tenants that were queried by different tenant_id values during the attack window, establishing the scope of data accessed | Docker container runtime logs for `dify-plugin-daemon` container (`docker logs dify-plugin-daemon --timestamps`): Internal API calls to Plugin Daemon that originated outside expected service-to-service communication patterns, surfacing exploitation of the unauthenticated internal API exposure described in the DifyTap disclosure

Per-Action IR Details

Step 1: Containment — Identify all Dify deployments in your environment (cloud-hosted and self-hosted). Restrict network access to Dify's API endpoints and Plugin Daemon interfaces at the perimeter firewall or WAF. For cloud-hosted Dify tenants, review access logs immediately for cross-tenant API calls or anomalous document retrieval requests. Disable external API access to Dify where not operationally required until patching is complete.

NIST Phase: Containment

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

Controls: NIST AC-4 (Information Flow Enforcement), NIST AC-17 (Remote Access), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory)

Compensating: On Linux self-hosted Dify instances, immediately apply iptables rules to whitelist only authorized source IPs to ports 3000 (Dify web/API) and 5002 (Plugin Daemon): `iptables -I INPUT -p tcp --dport 5002 -j DROP` followed by explicit ACCEPT rules for trusted CIDRs. Use `ss -tnp | grep -E '3000|5002'` to enumerate active connections to these ports before dropping. For cloud-hosted tenants where you cannot block at the network layer, use Dify's admin console to disable API key access temporarily.

Evidence: BEFORE restricting firewall/WAF rules or disabling API access, capture: (1) active TCP connection state on Dify host via `ss -tnp` or `netstat -ano` to record any live sessions to Plugin Daemon port 5002 — these connections

are destroyed the moment the firewall rule drops them; (2) Dify application API access logs (typically at `/var/log/dify/` or Docker container logs via `docker logs dify-api`) showing in-flight request URIs, tenant IDs, and source IPs; (3) for cloud-hosted tenants, export the last 72 hours of Dify audit/access logs before any account or API key changes are made, as log retention windows may be short.

Step 2: Detection — Query application and API gateway logs for requests to Dify's Plugin Daemon endpoints from users or service accounts outside expected tenant scopes. Look for path traversal patterns (e.g., '../' sequences) in API request URLs targeting document or file endpoints. Correlate against MITRE T1083 (file and directory discovery) and T1602 (data from configuration repositories) behavioral indicators. Review audit logs (NIST AU-6, CIS 8.2) for anomalous cross-tenant conversation retrieval.

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-2 (Event Logging), NIST AU-3 (Content Of Audit Records), CIS 8.2 (Collect Audit Logs)

Compensating: Without a SIEM, run the following directly against Dify's nginx/API access logs: `grep -E '(\.\./|%2e%2e%2f|%252e)' /var/log/nginx/access.log` to surface path traversal attempts targeting document endpoints. For cross-tenant anomalies, extract the `X-Tenant-ID` or equivalent header field from Dify API logs and run `awk '{print $tenant_field}' access.log | sort | uniq -c | sort -rn` to identify tenant IDs appearing in unexpected request contexts. Use `docker logs dify-api 2>&1 | grep -iE '(plugin.daemon|/v1/files|/v1/datasets)' | grep -v 200` to surface non-200 responses on sensitive endpoints that may indicate probing. Deploy a Sigma rule matching on URI patterns `*/files/*` and `*/datasets/*` with source tenant mismatch as a grep-based daily cron alert.

Evidence: This is an analytical phase — no live state is being altered. Capture and preserve: (1) full Dify API gateway/nginx access logs for the preceding 90 days (cross-tenant exfiltration may predate discovery), specifically request lines containing `/v1/files`, `/v1/datasets`, `/console/api`, and Plugin Daemon paths; (2) Dify application-level audit logs recording conversation retrieval events, including the tenant_id and user_id fields associated with each chat history access; (3) any WAF logs (OWASP ModSecurity, AWS WAF, Cloudflare) showing requests with `../` or URL-encoded equivalents (`%2e%2e%2f`, `%252e%252e%252f`) in paths targeting Dify document endpoints — these are the specific forensic signature of CVE-2026-41948 exploitation.

Step 3: Eradication — Upgrade all self-hosted Dify instances to v1.14.2 or later per the vendor release at https://github.com/langgenius/dify. For CVE-2026-41948 (path traversal, unpatched), apply WAF rules to block path traversal sequences in requests targeting Dify document/file endpoints as an interim control until the vendor issues a fix. Rotate API keys and session tokens for all Dify tenants as a precaution per D3-CRO (Credential Rotation).

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: For the Dify v1.14.2 upgrade on Docker-based self-hosted deployments: `docker pull langgenius/dify-api:1.14.2 && docker pull langgenius/dify-web:1.14.2`, then update `docker-compose.yml` image tags and run `docker-compose up -d`. Before upgrading, snapshot the running container state and any mounted volumes. For CVE-2026-41948 WAF mitigation without an enterprise WAF, add an OWASP ModSecurity CRS rule to nginx: `SecRule REQUEST_URI "@rx (?:%2e|\.)(?:%2e|\.)(?:%2f|/)" "id:10001,phase:1,deny,status:400,msg:'Path traversal blocked on Dify endpoint'"`. Credential rotation: use Dify admin UI to invalidate all API keys under Settings → API Keys, and force session expiry by restarting the Redis session store (`docker restart dify-redis`).

Evidence: BEFORE patching or rotating credentials, capture: (1) memory image or at minimum process memory strings from the running `dify-api` and `dify-plugin-daemon` containers using `docker exec dify-api cat /proc/1/maps` and `strings /proc/1/mem` where accessible — live memory may contain active session tokens and cross-tenant request payloads that are lost after restart; (2) a full export of current Dify API key inventory and associated tenant mappings from the Dify database (`SELECT * FROM api_tokens` against the Dify Postgres instance) to establish a pre-rotation baseline for forensic comparison; (3) a snapshot of all currently active sessions in Redis (`redis-cli KEYS

'session:*'` with value dumps) before the Redis restart invalidates them — these may contain evidence of attacker-held sessions.

Step 4: Recovery — After upgrading to v1.14.2, validate that authorization controls enforce tenant boundary isolation by testing cross-tenant API requests and confirming they are rejected. Review access control lists on document storage associated with Dify (NIST AC-3, CIS 3.3). Monitor Dify API logs for 30 days post-patch for residual anomalous access patterns. Confirm CVE-2026-41948 path traversal is blocked at the WAF layer until the vendor patch is released.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST AC-3 (Access Enforcement), CIS 3.3 (Configure Data Access Control Lists), NIST AU-6 (Audit Record Review, Analysis, And Reporting)

Compensating: Validate tenant boundary enforcement post-upgrade by creating two test Dify tenants and issuing API calls with Tenant A's credentials attempting to retrieve Tenant B's conversation history and uploaded documents via `/v1/messages`, `/v1/files`, and `/v1/datasets` endpoints — all should return HTTP 403. For document storage ACL review, run `ls -la` on Dify's local file storage path (default: `volumes/app/storage/` in Docker deployments) and verify that uploaded documents are namespaced under per-tenant directories with no world-readable permissions (`chmod 750` at minimum). For 30-day monitoring without SIEM, schedule a daily cron: `grep -E '(HTTP/[0-9.]+ [45][0-9]{2})' /var/log/nginx/access.log | grep -E '(/v1/files|/v1/datasets|/v1/messages)' >> /var/log/dify_anomaly_review.log`.

Evidence: Recovery validation is a low-volatility phase, but preserve: (1) output of the cross-tenant authorization test calls as a timestamped artifact proving boundary enforcement is restored — specifically HTTP response codes and bodies from the test requests described above; (2) a post-upgrade snapshot of Dify database `tenant_id` foreign key relationships on the `conversations`, `messages`, and `uploaded_files` tables to confirm no orphaned cross-tenant references persist from the attack window; (3) WAF block logs showing CVE-2026-41948 path traversal attempts being rejected after rule deployment, as evidence of compensating control effectiveness for the unpatched vulnerability.

Step 5: Post-Incident — Conduct a review of authorization architecture for all AI/ML platforms in your environment, specifically evaluating tenant isolation controls and internal API exposure (NIST AC-4, AC-6). Evaluate whether Plugin Daemon and other internal service APIs are accessible from external network segments and remediate as needed. Document control gaps and update your AI platform security baseline. Consider adding D3-LAM (Local Account Monitoring) and D3-UAP (User Account Permissions) reviews to your AI platform hardening checklist.

NIST Phase: Post Incident

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

Controls: NIST AC-4 (Information Flow Enforcement), NIST AC-6 (Least Privilege), CIS 4.2 (Establish and Maintain a Secure Configuration Process for Network Infrastructure), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Conduct a network exposure review of all AI/ML platform internal APIs using nmap: `nmap -sV -p 5002,3000,8080,11434 --open ` to discover Plugin Daemon and similar internal service ports reachable from network segments they should not serve. For AI platform authorization architecture review, build a simple matrix in a spreadsheet mapping each platform's API endpoints to their authentication requirement (none / API key / session token / MFA), tenant isolation mechanism, and whether internal-only APIs are routable from internet-facing segments — the DifyTap disclosure shows Plugin Daemon (port 5002) was reachable externally, which should be the template for what to check across other platforms.

Evidence: Post-incident documentation artifacts to preserve: (1) the full incident timeline reconstructed from Dify API logs, WAF logs, and Redis session data showing the earliest evidence of cross-tenant access attempts against CVE-2024-5846, CVE-2026-41947, CVE-2026-41948, CVE-2026-41949, and CVE-2026-41950 exploitation; (2) a network topology diagram or firewall rule export showing the pre-incident exposure of Dify Plugin Daemon port 5002 to untrusted network segments, as evidence for the lessons-learned report; (3) an inventory of all AI/ML platform tenants and the specific conversation histories and uploaded documents potentially accessible during the exposure window, to support any breach notification assessment for PII or confidential business data exfiltrated via the cross-tenant flaws.

Detection Guidance

Focus detection on three behavioral patterns. First, cross-tenant authorization bypass: query API gateway and application logs for requests where the authenticated user's tenant identifier does not match the tenant scope of the resource being accessed, flag any 200-level responses to such requests in Dify's conversation or document endpoints. Second, Plugin Daemon traversal: monitor HTTP request logs for path traversal sequences ('../', '%2e%2e%2f', '%252e%252e%2f') in URLs targeting Dify's internal Plugin Daemon API routes. Third, anomalous document exfiltration: alert on bulk or rapid sequential document retrieval requests from a single session or low-reputation source IP, particularly to Dify's file/upload endpoints (T1119, T1530, T1567). Relevant NIST controls: AU-2 (event logging), AU-6 (audit record review), SI-4 (system monitoring). CIS 8.2 (collect audit logs) should be verified as enabled for all Dify API surfaces. No confirmed IOCs (IPs, domains, hashes) have been published in available source material.

Framework Mappings

MITRE-ATTACK

- **T1552** — Unsecured Credentials
- **T1083** — File and Directory Discovery
- **T1602** — Data from Configuration Repository
- **T1119** — Automated Collection
- **T1190** — Exploit Public-Facing Application
- **T1078** — Valid Accounts
- **T1567** — Exfiltration Over Web Service
- **T1530** — Data from Cloud Storage

NIST-800-53R5

- **AC-6** — Least Privilege
- **SC-7** — Boundary Protection
- **SI-4** — System Monitoring
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SI-2** — Flaw Remediation
- **SI-7** — Software, Firmware, and Information Integrity
- **AC-2** — Account Management
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **SI-16** — Memory Protection
- **AC-3** — Access Enforcement
- **SI-10** — Information Input Validation

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures
- **16.12** — Implement Code-Level Security Checks
- **6.1** — Establish an Access Granting Process
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

OWASP-TOP10-2021

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

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.23** — Information security for use of cloud services

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1552	Unsecured Credentials	Credential-Access
T1083	File and Directory Discovery	Discovery
T1602	Data from Configuration Repository	Collection
T1119	Automated Collection	Collection
T1190	Exploit Public-Facing Application	Initial-Access
T1078	Valid Accounts	Defense-Evasion
T1567	Exfiltration Over Web Service	Exfiltration
T1530	Data from Cloud Storage	Collection

Sources

Source	URL	Tier
Security News	https://thehackernews.com/2026/06/researchers-detail-difytap-flaws-...	T3
CVE-2026-41950 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-41950	T1
Vulnerability Signatures in 2026 - Palo Alto Networks | TechDocs	https://docs.paloaltonetworks.com/iot/release-notes/vulnerability-s...	T3

Source	URL	Tier
CVEs and Security Vulnerabilities - OpenCVE	https://app.opencve.io/	T3
CVE Record: CVE-2026-46950 - Oracle	https://www.cve.org/CVERecord?id=CVE-2026-46950	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2024-5846, CVE-2026-41947, CVE...	T1

Generated 2026-06-22 19:03 UTC by TJS Security Command Center