INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-23 06:27 UTC

CVE-2026-9082: Drupal Core SQL Injection Under Active Attack Within 48 Hours of Patch Release

CVE VULNERABILITY | CRITICAL | CVSS 9.5

SCC Item ID	SCC-CVE-2026-0212
Type	CVE Vulnerability
CVE ID	CVE-2026-9082
Severity	CRITICAL
CVSS Base Score	9.5
EPSS Score	0.0002 (5th percentile)
Affected Products	Drupal Core 8.9, 9.5, 10.4.x, 10.5.x, 10.6.x, 11.1.x, 11.2.x, 11.3.x
Published	2026-05-23T03:23:48
Discovery Source	Rss

Executive Summary

A critical SQL injection flaw in Drupal Core (CVE-2026-9082, CVSS 9.5) moved from patch release to confirmed active exploitation in under 48 hours. According to vendor telemetry reports, attackers have made over 15,000 documented attempts against approximately 6,000 sites across 65 countries, with gaming and financial services organizations accounting for roughly half of observed targets. Current attack patterns indicate adversaries are in active reconnaissance and initial access phases, meaning organizations running unpatched Drupal instances have a narrowing window to remediate before data extraction or privilege escalation attempts begin.

Technical Analysis

CVE-2026-9082 is a critical SQL injection vulnerability (CWE-89) with a privilege escalation component (CWE-269) in Drupal Core, carrying a CVSS base score of 9.5. Affected versions span Drupal 8.9, 9.5, 10.4.x, 10.5.x, 10.6.x, 11.1.x, 11.2.x, and 11.3.x. The flaw is exploitable via the web interface (T1190, Exploit Public-Facing Application) without authentication requirements confirmed in available sourcing. Post-exploitation potential techniques include file and directory discovery (T1083), network service scanning (T1046), command and scripting interpreter abuse (T1059), and valid account leverage (T1078), consistent with initial access and enumeration tradecraft; confirmation of these techniques in active exploitation requires ongoing threat intelligence monitoring. Vendor telemetry reports over 15,000 attack attempts targeting approximately 6,000 sites across 65 countries within the first 48 hours post-patch. Attribution is unestablished. CISA KEV status

should be verified against the live CISA catalog before citing in reporting. Source URLs reflect 2026 publication dates (current year). URLs should be verified against live sources before republication.

Action Checklist

1. Step 1: Containment, Identify all internet-facing Drupal instances running versions 8.9 (unsupported), 9.5 (unsupported), 10.4.x, 10.5.x, 10.6.x, 11.1.x, 11.2.x, or 11.3.x using your asset inventory (CIS 1.1, Enterprise Asset Inventory). Place WAF rules blocking SQL injection patterns against Drupal endpoints immediately. If immediate patching is not feasible, restrict public access to affected Drupal admin paths and database-facing endpoints. Apply NIST AC-17 (Remote Access) controls to limit exposure of management interfaces.
2. Step 2: Detection, Query web application logs and WAF telemetry for anomalous SQL syntax in request parameters targeting Drupal routes (look for UNION SELECT, OR 1=1, stacked queries, and encoded variants). Review database query logs for unexpected SELECT, INSERT, or UPDATE statements originating from the Drupal application account. Check authentication logs for unexpected privilege escalation events or new account creation (NIST AU-6, Audit Record Review; CIS 8.2, Collect Audit Logs). Correlate against the MITRE ATT&CK techniques T1046 (network scanning from compromised host), T1083 (file enumeration), and T1059 (script execution) for post-exploitation indicators.
3. Step 3: Eradication, Apply the official Drupal security patch for CVE-2026-9082 per the Drupal Security Team advisory at https://www.drupal.org/security. Verify the specific patch version for your branch against the published advisory. For Drupal 8.9 and 9.5 (unsupported branches), migration to a supported version is required; no patch may be available for end-of-life branches. After patching, rotate all database credentials and application service account passwords used by the affected Drupal instance (NIST AC-2, Account Management).
4. Step 4: Recovery, Validate patch application by confirming the installed Drupal version matches the patched release. Review database tables for unauthorized records, unexpected user accounts, or modified content (NIST SI-4, System Monitoring). Re-enable production traffic only after WAF rules are confirmed active and log collection is verified (NIST AU-2, Event Logging). Monitor for recurrence of SQL injection patterns and any privilege escalation attempts for a minimum of 72 hours post-remediation. Conduct a focused review of privileged account activity for the exploitation window.
5. Step 5: Post-Incident, Document the time between patch release and your organization's remediation completion; if it exceeded 48 hours, assess your patch prioritization process against CIS 7.1 (Vulnerability Management Process) and CIS 7.2 (Remediation Process). Review whether internet-facing Drupal instances were included in your asset inventory and vulnerability scanning scope (CIS 1.1, CIS 7.3). Evaluate whether WAF coverage existed for Drupal endpoints prior to exploitation; gaps here indicate a control deficiency against NIST SI-4. Conduct a privilege review to enforce least privilege on database accounts used by Drupal (NIST AC-6, Least Privilege).

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO and legal/compliance immediately if forensic review of `users_field_data`, `file_managed`, or MySQL query logs confirms unauthorized data access, account creation, or webshell deployment — confirmed exploitation of CVE-2026-9082 on a site handling PII or financial data triggers breach notification obligations under GDPR, CCPA, and PCI DSS §12.10, and CISA KEV listing creates additional federal reporting obligations for applicable entities.
Recovery Notes	Before re-enabling production traffic, verify the patched Drupal version via `drush status`, confirm WAF SQLi rules are blocking test payloads against the Drupal endpoints, and validate that all database credentials rotated in Step 3 are reflected in `settings.php` with no cached copies in PHP opcache (`php -r 'opcache_reset();'`). Maintain heightened log monitoring on web application and database query logs for a minimum of 72 hours post-patch, specifically watching for `UNION SELECT`, `information_schema`, and `INTO OUTFILE` patterns that would indicate re-exploitation or a persistent threat actor retesting after remediation. Given the confirmed 15,000+ documented attack attempts across 6,000 sites, treat any post-remediation SQLi attempt sourced from a previously observed attacker IP as evidence of targeted follow-on activity rather than opportunistic scanning.
Forensic Artifacts	Nginx/Apache access logs filtered for Drupal route parameters containing URL-encoded SQL metacharacters (%27, %3b, %20UNION%20, %20OR%201%3d1) — these are the direct fingerprint of CVE-2026-9082 exploitation attempts against Drupal's front-controller | MySQL general query log entries executed under the Drupal application database account referencing `information_schema.tables`, `information_schema.columns`, `LOAD_FILE()`, or `SELECT ... INTO OUTFILE` — indicating successful SQLi payload execution beyond the reconnaissance phase | Drupal `watchdog` table (exportable via `drush watchdog-show`) for PDOException or DatabaseExceptionWrapper entries timestamped within the 48-hour active exploitation window, which Drupal logs when malformed SQL queries cause database errors during injection probing | Filesystem artifacts: `.php` files written outside `/core`, `/modules`, `/themes`, and `/vendor` directories — particularly in `/sites/default/files/` which is web-accessible and the canonical webshell drop location for post-SQLi `INTO OUTFILE` persistence on Drupal installations | Drupal `users_field_data` and `user__roles` database tables diffed against a known-good backup, specifically looking for `administrator`-role accounts created or modified during the exploitation window via direct DB INSERT rather than the Drupal admin UI — a direct indicator of SQLi-driven privilege escalation

Per-Action IR Details

Step 1: Containment — Identify all internet-facing Drupal instances running versions 8.9, 9.5, 10.4.x, 10.5.x, 10.6.x, 11.1.x, 11.2.x, or 11.3.x using your asset inventory (CIS 1.1 — Enterprise Asset Inventory). Place WAF rules blocking SQL injection patterns against Drupal endpoints immediately. If immediate patching is not feasible, restrict public access to affected Drupal admin paths and database-facing endpoints. Apply NIST AC-17 (Remote Access) controls to limit exposure of management interfaces.

NIST Phase: Containment

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

Controls: NIST AC-17 (Remote Access), NIST AC-4 (Information Flow Enforcement), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Run `curl -s https:///CHANGELOG.txt` or check `/core/lib/Drupal.php` for the VERSION constant to enumerate Drupal versions across hosts without a CMDB. Deploy ModSecurity with the OWASP CRS ruleset (free) and enable paranoia level 2 rules targeting SQL injection — specifically rules 942100–942999 — against all Drupal `.php` endpoints. Block admin paths `/admin`, `/user/login`, and `/node/add` at the network perimeter using iptables or an nginx `deny all` directive scoped to those URI prefixes.

Evidence: Before isolating any instance, capture the current active HTTP connection table (`ss -tnp` or `netstat -antp`) to document source IPs actively interacting with Drupal PHP processes. Snapshot WAF deny/allow logs with timestamps covering the prior 48-hour window since patch release — this is the confirmed active exploitation window. Export the Drupal `users_field_data` and `sessions` database tables immediately to establish a baseline of accounts and active sessions prior to any access restriction changes.

Step 2: Detection — Query web application logs and WAF telemetry for anomalous SQL syntax in request parameters targeting Drupal routes (look for UNION SELECT, OR 1=1, stacked queries, and encoded variants). Review database query logs for unexpected SELECT, INSERT, or UPDATE statements originating from the Drupal application account. Check authentication logs for unexpected privilege escalation events or new account creation (NIST AU-6 — Audit Record Review; CIS 8.2 — Collect Audit Logs). Correlate against the MITRE ATT&CK techniques T1046 (network scanning from compromised host), T1083 (file enumeration), and T1059 (script execution) for post-exploitation indicators.

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 SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Parse Apache/Nginx access logs with this grep to surface encoded and plaintext SQLi against Drupal routes: `grep -iE '(union.+select|or.+1=1|sleep\(|benchmark\(|%27|%3b|information_schema|0x[0-9a-f]+)' /var/log/apache2/access.log`. Enable MySQL/MariaDB general query log temporarily (`SET GLOBAL general_log = 'ON'; SET GLOBAL general_log_file='/var/log/mysql/general.log';`) and filter for queries issued by the Drupal DB user: `grep -i '' /var/log/mysql/general.log | grep -iE '(union|information_schema|load_file|outfile)'`. Use the free Sigma rule `web_drupal_sqli_cve_2026_9082` pattern (or draft one targeting Drupal route parameters) with `sigmac` converted to a grep/awk pipeline if no SIEM is available.

Evidence: Collect the full Nginx or Apache access log covering the 48-hour post-patch window and filter on Drupal's front-controller path (`/index.php` and clean URLs) for requests containing SQL metacharacters in GET/POST parameters — these are the injection points for CVE-2026-9082. Pull the MySQL slow query log and general query log for statements executed under the Drupal application database account that reference `information_schema.tables`, `LOAD_FILE`, or `INTO OUTFILE` — hallmarks of SQLi data extraction. Dump the Drupal `watchdog` table (`drush watchdog-show --count=500 --severity=0`) for PHP errors, database exceptions, or access-denied entries generated during exploitation attempts, as Drupal logs failed DB calls to its own watchdog before they surface in system logs.

Step 3: Eradication — Apply the official Drupal security patch for CVE-2026-9082 per the Drupal Security Team advisory. Upgrade affected branches to the patched release: verify the specific patch version against the Drupal security advisories page (https://www.drupal.org/security — human verification recommended). For Drupal 8.9 and 9.5 (unsupported branches), migration to a supported version is required; no patch may exist for end-of-life branches. After patching, rotate all database credentials and application service account passwords used by the affected Drupal instance (D3-CRO — Credential Rotation; NIST AC-2 — Account Management).

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST AC-2 (Account Management), NIST CM-6 (Configuration Settings), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 5.2 (Use Unique Passwords)

Compensating: For Drupal 10.x/11.x, use Composer to apply the patch: `composer require drupal/core-recommended: --update-with-dependencies && drush updatedb && drush cr`. For Drupal 8.9 and 9.5 (EOL — no patch exists), immediately place the site in maintenance mode (`drush sset system.maintenance_mode 1`) and block all public HTTP access via firewall rule as a hard compensating control until migration to a supported branch is complete. Rotate the Drupal DB credential by updating `settings.php` (`$databases['default']['default']['password']`) and simultaneously revoking the old credential in MySQL: `ALTER USER 'drupal_user'@'localhost' IDENTIFIED BY '';`.

Evidence: Before applying the patch, image or snapshot the webroot (`tar -czf drupal_webroot_preremediaton_$(date +%Y%m%d).tar.gz /var/www/html`) and the database (`mysqldump --single-transaction drupal_db > drupal_db_preremediation_$(date +%Y%m%d).sql`) to preserve forensic state for post-incident analysis. Capture the current contents of `sites/default/settings.php` and any `settings.local.php` — a successful SQLi exploitation of CVE-2026-9082 could have been used to read or exfiltrate the database credentials stored there. Document the installed module list via `drush pm-list --status=enabled` before patching, as attackers may have installed malicious modules via the compromised DB to establish persistence.

Step 4: Recovery — Validate patch application by confirming the installed Drupal version matches the patched release. Review database tables for unauthorized records, unexpected user accounts, or modified content (NIST SI-4 — System Monitoring). Re-enable production traffic only after WAF rules are confirmed active and log collection is verified (NIST AU-2 — Event Logging). Monitor for recurrence of SQL injection patterns and any privilege escalation attempts for a minimum of 72 hours post-remediation. Conduct a focused review of privileged account activity for the exploitation window (D3-LAM — Local Account Monitoring).

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST SI-4 (System Monitoring), NIST AU-2 (Event Logging), NIST AC-6 (Least Privilege), NIST CM-6 (Configuration Settings), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Confirm patch version with: `drush status | grep 'Drupal version'` and cross-reference against the Drupal security advisory (human-verify at https://www.drupal.org/security). Query the `users_field_data` table for accounts created or modified during the exploitation window: `SELECT uid, name, mail, created, changed FROM users_field_data WHERE created >= UNIX_TIMESTAMP('') ORDER BY created DESC;`. Deploy a lightweight Sigma-to-grep watchdog script running every 5 minutes via cron that tails the Nginx/Apache access log and alerts (email or local log) on any recurrence of the CVE-2026-9082 SQLi patterns for the mandated 72-hour monitoring window.

Evidence: Compare the live `users_field_data` and `user__roles` tables against your pre-remediation database snapshot to identify any accounts granted `administrator` role during the exploitation window — SQLi exploitation of Drupal Core at CVSS 9.5 commonly enables direct DB-level privilege escalation. Audit the Drupal file system for newly written `.php` files outside the expected module/theme directories using `find /var/www/html -name '*.php' -newer /var/www/html/core/lib/Drupal.php -not -path '*/cache/*'` — SQLi with `INTO OUTFILE` capability can drop webshells. Review the `file_managed` and `node__body` database tables for injected content or unexpected file records created during the exploitation window.

Step 5: Post-Incident — Document the time between patch release and your organization's remediation completion; if it exceeded 48 hours, assess your patch prioritization process against CIS 7.1 (Vulnerability Management Process) and CIS 7.2 (Remediation Process). Review whether internet-facing Drupal instances were included in your asset inventory and vulnerability scanning scope (CIS 1.1, CIS 7.3). Evaluate whether WAF coverage existed for Drupal endpoints prior to exploitation — gaps here indicate a control deficiency against NIST SI-4. Conduct a privilege review to enforce least privilege on database accounts used by Drupal (NIST AC-6 — Least Privilege; D3-UAP — User Account Permissions).

NIST Phase: Post Incident

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

Controls: NIST AC-6 (Least Privilege), NIST SI-2 (Flaw Remediation), NIST AU-11 (Audit Record Retention), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 7.3 (Perform Automated Operating System Patch Management)

Compensating: Measure your patch cycle time by diffing the Drupal security advisory publication timestamp against your Composer lock file commit timestamp in version control (`git log --follow composer.lock | head -20`). Audit the Drupal DB user's MySQL privileges with `SHOW GRANTS FOR 'drupal_user'@'localhost';` — the account should hold

only SELECT, INSERT, UPDATE, DELETE on the Drupal database, never FILE, SUPER, or GRANT OPTION, which would have amplified SQLi impact. Add Drupal's security advisory RSS feed (https://www.drupal.org/security/rss.xml — human-verify) to your vulnerability intake process so future critical advisories are ingested within hours, not days.

Evidence: Retain the full Nginx/Apache access logs, MySQL general query logs, Drupal watchdog export, and pre-remediation database and webroot snapshots for a minimum of 90 days to support regulatory breach notification analysis — given observed targeting of financial services organizations, these logs may be required for PCI DSS incident documentation. Document all source IPs that sent confirmed SQLi payloads during the exploitation window and submit as IOCs to your threat intelligence platform or share via ISAC if you are a member of FS-ISAC or the Gaming/Hospitality ISAC, given the sector-specific targeting pattern of this campaign.

Detection Guidance

Primary detection surface is web application and database logs. Query WAF and web server access logs for requests to Drupal endpoints containing SQL metacharacters: single quotes, double dashes, UNION SELECT, OR 1=1, semicolons in parameter values, and URL-encoded equivalents (%27, %3B, %2D%2D). Filter for HTTP 200 or 500 responses to these requests; 200 may indicate successful injection, 500 may indicate probing. Review database slow query logs and general query logs for SELECT statements containing UNION clauses or subqueries not matching known application patterns. Monitor for new database user creation or permission grants issued by the Drupal application database account. Per NIST AU-6, establish a review cadence for these log sources at least daily during the active exploitation window. For behavioral indicators consistent with post-exploitation: watch for unusual outbound connections from the web server (T1046, Network Service Scanning), unexpected script execution from the web root (T1059), and file enumeration activity in application directories (T1083). CIS 8.2 requires audit log collection to be active across enterprise assets; confirm Drupal application logging and database query logging are enabled before relying on absence of evidence. No confirmed IOCs (IPs, hashes, domains) were available in the ingested sourcing; monitor CISA KEV, Drupal security advisories, and threat intelligence feeds for IOC updates as the campaign matures.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`No confirmed IOCs available in ingested sourcing`	Attack telemetry (Imperva) documents volume and geographic distribution but no specific IPs, domains, or hashes were included in available source data. Monitor CISA KEV and threat intelligence feeds for IOC updates.	LOW

Framework Mappings

MITRE-ATTACK

- **T1083** — File and Directory Discovery
- **T1190** — Exploit Public-Facing Application
- **T1078** — Valid Accounts
- **T1046** — Network Service Discovery

- **T1059** — Command and Scripting Interpreter

NIST-800-53R5

- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **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
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-10** — Information Input Validation
- **IR-5** — Incident Monitoring

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control
- **A03:2021** — Injection

CIS-V8

- **5.4** — Restrict Administrator Privileges to Dedicated Administrator Accounts
- **6.8** — Define and Maintain Role-Based Access Control
- **16.10** — Apply Secure Design Principles in Application Architectures
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

ISO-27001-2022

- **A.8.28** — Secure coding
- **A.8.8** — Management of technical vulnerabilities

HIPAA-SECURITY

- **164.312(e)(1)** — Transmission Security

NIST-CSF-2

- **DE.AE-08** — Incidents are declared when adverse events meet the defined incident criteria

SOC2-TSC

- **CC6.3** — Authorizes, modifies, or removes access

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1083	File and Directory Discovery	Discovery
T1190	Exploit Public-Facing Application	Initial-Access
T1078	Valid Accounts	Defense-Evasion
T1046	Network Service Discovery	Discovery
T1059	Command and Scripting Interpreter	Execution

Sources

Source	URL	Tier
Security News	https://thehackernews.com/2026/05/drupal-core-sql-injection-bug-act...	T3
CVE-2026-9082: Critical Drupal Core SQLi Flaw - SOC Prime	https://socprime.com/blog/cve-2026-9082-analysis/	T3
CVE-2026-9082 | Tenable®	https://www.tenable.com/cve/CVE-2026-9082	T3
Drupal Core SQL Injection Vulnerability CVE-2026-9082	https://security.berkeley.edu/news/drupal-core-sql-injection-vulner...	T1
CVE-2026-9082 - TuxCare Vulnerabilities	https://tuxcare.com/cve-tracker/cve/details/cve-2026-9082/	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-9082	T1

Generated 2026-05-23 06:27 UTC by TJS Security Command Center