INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-22 06:23 UTC

pgAdmin 4, Multiple Critical Vulnerabilities (XSS, CSRF, SQLi, AI Assistant Bypass)

CVE VULNERABILITY | CRITICAL | CVSS 9.0

SCC Item ID	SCC-CVE-2026-0342
Type	CVE Vulnerability
CVE ID	CVE-2026-12048, CVE-2026-12046, CVE-2026-12045, CVE-2026-12044
Severity	CRITICAL
CVSS Base Score	9.0
Affected Products	pgAdmin 4 < 9.16 (PostgreSQL administration and development platform)
Published	2026-06-20
Discovery Source	Gemini

Executive Summary

Four critical vulnerabilities in pgAdmin 4 (versions prior to 9.16) expose PostgreSQL database administration environments to stored cross-site scripting, cross-site request forgery, SQL injection, and an AI Assistant bypass that can enable unauthorized write operations against managed databases. Any organization running pgAdmin 4 as a database management interface should treat this as a priority patching event, as successful exploitation could allow attackers to execute arbitrary code in administrator browsers, manipulate database queries, and compromise session integrity. The combination of vulnerabilities in a single administrative tool used to manage PostgreSQL infrastructure elevates business risk to data integrity, availability, and confidentiality across all databases administered through the platform.

Technical Analysis

pgAdmin 4 versions prior to 9.16 contain four critical vulnerabilities reported in a vendor advisory. CVE-2026-12048 (CWE-79) is a stored XSS flaw allowing persistent arbitrary JavaScript execution in the browsers of authenticated administrators. CVE-2026-12046 (CWE-352) reflects absent CSRF token enforcement, permitting forged authenticated requests that can alter user editor state without user consent. CVE-2026-12045 (CWE-863) is a read-only transaction bypass in the pgAdmin AI Assistant feature; it may allow attackers to coerce write operations against connected PostgreSQL instances that should be restricted to read-only contexts. CVE-2026-12044 (CWE-89) is a SQL injection flaw enabling manipulation of database queries through unsanitized input. MITRE ATT&CK techniques mapped: T1190 (Exploit Public-Facing

Application), T1185 (Browser Session Hijacking), T1565.001 (Stored Data Manipulation), T1059.007 (Command and Scripting Interpreter: JavaScript). CVSS base is 9.0 (critical). Official CVSS vector from NVD is pending publication; this base score reflects vendor advisory severity assessment. Remediation: upgrade to pgAdmin 4 version 9.16 or later. No CISA KEV listing confirmed at this time. Sources include NVD detail pages, Tenable plugin pipeline (issue 222825), and threat-modeling.com intelligence reporting dated June 20, 2026.

Action Checklist

1. Step 1: Containment, Identify all instances of pgAdmin 4 prior to version 9.16 across your environment. If internet-facing, restrict access immediately using firewall rules or WAF policies to limit exposure to trusted IP ranges only (NIST AC-17, CIS 4.4). Do not leave unauthenticated or publicly routable pgAdmin interfaces accessible during the remediation window.
2. Step 2: Detection, Review web server and application logs for pgAdmin 4 for anomalous JavaScript payloads in input fields (XSS indicators), unexpected POST requests missing CSRF tokens, unusual AI Assistant query patterns with write-intent syntax, and SQL error messages or unexpected query structures suggesting injection attempts. Audit active sessions for signs of session hijacking consistent with T1185. Reference AU-6 (Audit Record Review) and CIS 8.2 (Collect Audit Logs) to ensure logging is active and being reviewed.
3. Step 3: Eradication, Upgrade all pgAdmin 4 installations to version 9.16 or later per the vendor advisory. Validate the upgrade across all deployment modes (desktop, server, container). After patching, rotate all database credentials accessible through pgAdmin sessions that may have been exposed, per NIST IA-5 (Authentication). Review AI Assistant feature configuration and disable it if not required.
4. Step 4: Recovery, After upgrading, verify the pgAdmin version reported in the application matches 9.16 or later. Re-enable access for legitimate users and confirm CSRF protections are active by reviewing application behavior. Monitor PostgreSQL database audit logs for any unauthorized write operations, schema changes, or anomalous queries that may have occurred during the exposure window, consistent with AU-6 and AU-11 (Audit Record Retention).
5. Step 5: Post-Incident, Conduct a control gap review against NIST SI-4 (System Monitoring) and CIS 7.1 (Establish and Maintain a Vulnerability Management Process) to assess whether pgAdmin 4 was included in your asset inventory and patch cadence. Evaluate whether AI-integrated features in administrative tools receive security review before deployment. Document lessons learned regarding administrative tool exposure and access restriction policies (NIST AC-17, AC-6).

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to CISO and legal/compliance immediately if PostgreSQL audit logs show unauthorized DDL operations (CREATE, DROP, ALTER), data exfiltration indicators (large SELECT result sets to external IPs), or if pgAdmin was accessible without authentication during the exposure window and the environment stores PII, PHI, or financial data subject to GDPR, HIPAA, or PCI-DSS breach notification requirements.

Recovery Notes	After upgrading to pgAdmin 9.16 and rotating all PostgreSQL credentials exposed through pgAdmin sessions, conduct a 30-day heightened monitoring period on PostgreSQL audit logs for anomalous query patterns — specifically DDL changes, privilege escalations ('GRANT', 'ALTER ROLE'), and bulk SELECT operations that could indicate delayed exfiltration from a prior AI Assistant bypass (CVE-2026-12044) or SQLi (CVE-2026-12045) compromise. Verify that any pgAdmin container deployments have been redeployed from the 9.16 image rather than in-place upgraded, as container layer caching may preserve vulnerable binaries. Confirm CSRF token validation is enforced end-to-end by reviewing the pgAdmin 9.16 release notes for the specific CSRF fix (CVE-2026-12046) and validating that the corrected token-check middleware is present in the deployed codebase.
Forensic Artifacts	pgAdmin 4 SQLite storage database (~/.pgadmin/pgadmin4.db or %APPDATA%\pgAdmin 4\pgadmin4.db): contains saved server connection definitions including encrypted or cleartext PostgreSQL credentials — a SQLi exploit via CVE-2026-12045 targeting this database would leave evidence of unauthorized read access in file system access timestamps and pgAdmin query logs | pgAdmin application log (~/.pgadmin/pgadmin4.log or /var/log/pgadmin/pgadmin4.log): records all user actions, AI Assistant queries, and error output — AI Assistant bypass (CVE-2026-12044) would appear as write-intent SQL (INSERT/UPDATE/DELETE/DROP) submitted through the AI query endpoint with no corresponding legitimate user session | PostgreSQL server log (/var/log/postgresql/postgresql--main.log with log_min_duration_statement and log_statement = 'all'): captures all queries executed against managed databases — successful SQLi (CVE-2026-12045) would produce syntax error sequences followed by successful UNION or stacked query execution from the pgAdmin service account | Web server access log (Apache /var/log/apache2/access.log or Nginx /var/log/nginx/access.log for server-mode deployments): stored XSS (CVE-2026-12048) delivery would appear as POST requests containing encoded JavaScript payloads (e.g., %3Cscript%3E, javascript:, onerror=) to pgAdmin input endpoints; CSRF exploitation (CVE-2026-12046) would appear as POST requests with anomalous or absent Referer headers and missing CSRF token parameters | pgAdmin sessions directory (~/.pgadmin/sessions/ or the configured SESSION_DB_PATH): contains active session token files — session hijacking following a successful stored XSS exploit (CVE-2026-12048) would be evidenced by the same session token appearing from multiple distinct source IPs within a short time window, visible by correlating session file modification timestamps against web server access logs

Per-Action IR Details

Step 1: Containment — Identify all instances of pgAdmin 4 prior to version 9.16 across your environment. If internet-facing, restrict access immediately using firewall rules or WAF policies to limit exposure to trusted IP ranges only (NIST AC-17, CIS 4.4). Do not leave unauthenticated or publicly routable pgAdmin interfaces accessible during the remediation window.

NIST Phase: Containment

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

Controls: NIST AC-17 (Remote Access), NIST AC-4 (Information Flow Enforcement), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices)

Compensating: Run 'netstat -tlnp | grep 5050' (default pgAdmin server port) or 'ss -tlnp | grep pgadmin' on each Linux host to identify exposed instances; on Windows use 'netstat -ano | findstr :5050'. Cross-reference against your CIS 1.1 asset inventory or run an nmap scan ('nmap -p 5050,443,80 --open ') to find unexpected pgAdmin listeners. Apply iptables or Windows Firewall rules scoped to trusted admin IPs immediately: 'iptables -I INPUT -p tcp --dport 5050 ! -s -j DROP'.

Evidence: Before restricting network access, capture active TCP connection state from each pgAdmin host ('ss -tnp' or 'Get-NetTCPConnection | Where-Object State -eq Established') to preserve a record of any live attacker sessions. Export the pgAdmin 4 session database ('%APPDATA%\pgAdmin 4\sessions\' on Windows or '~/.pgadmin/sessions/' on Linux) and the pgAdmin server log ('~/.pgadmin/pgadmin4.log' or the configured log path) to document any session tokens active at containment time. Firewall rule changes do not alter disk state but will sever live connections, destroying TCP session metadata — capture first.

Step 2: Detection — Review web server and application logs for pgAdmin 4 for anomalous JavaScript payloads in input fields (XSS indicators), unexpected POST requests missing CSRF tokens, unusual AI Assistant query patterns with write-intent syntax, and SQL error messages or unexpected query structures suggesting injection attempts. Audit active sessions for signs of session hijacking consistent with T1185. Reference AU-6 (Audit Record Review) and CIS 8.2 (Collect Audit Logs) to ensure logging is active and being reviewed.

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: Parse the pgAdmin application log for XSS indicators using grep: 'grep -iE "-main.log') for syntax error patterns: 'grep -i "syntax error\|pg_sleep\|union select\|or 1=1" /var/log/postgresql/*.log'. For AI Assistant abuse, filter pgAdmin logs for write-intent keywords (INSERT, UPDATE, DELETE, DROP, CREATE) originating from the AI query endpoint ('/pgadmin4/sql/query_tool/sql').

Evidence: This is a detection/analysis step that does not alter live state. Preserve the following before any containment actions invalidate them: (1) pgAdmin 4 application log in its current state ('~/.pgadmin/pgadmin4.log' or the server-mode equivalent under '/var/log/pgadmin/'); (2) PostgreSQL server logs showing query execution history including any anomalous DDL/DML operations tied to the pgAdmin session user; (3) web server access logs (Apache/Nginx) capturing full URI, POST body size, response codes, and client IPs for the pgAdmin vhost; (4) active pgAdmin session cookies from the sessions directory to correlate with any hijacked session indicators; (5) network capture (tcpdump -i any -w pgadmin_capture.pcap port 5050) if the interface is still reachable, to record in-flight payloads.

Step 3: Eradication — Upgrade all pgAdmin 4 installations to version 9.16 or later per the vendor advisory. Validate the upgrade across all deployment modes (desktop, server, container). After patching, rotate all database credentials accessible through pgAdmin sessions that may have been exposed, per D3-CRO (Credential Rotation). Review AI Assistant feature configuration and disable it if not required.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST AC-2 (Account Management), 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 desktop mode, verify the installed version via 'pip show pgadmin4' (Python/pip install) or check the pgAdmin About dialog. For server/container mode, confirm the running image tag: 'docker inspect | grep -i pgadmin' and compare against the official pgAdmin Docker Hub tag for 9.16. Credential rotation for PostgreSQL roles accessible through pgAdmin: connect directly via psql and execute 'ALTER ROLE WITH PASSWORD \'\';' for each affected role. To disable the AI Assistant in pgAdmin 9.16 server mode, set 'AI_SQL_GENERATOR_ENABLED = False' in config_local.py and restart the service.

Evidence: Before applying the pgAdmin 9.16 upgrade and before rotating credentials, capture the following volatile state: (1) acquire a full memory dump of the pgAdmin server process (using 'procdump -ma ' on Windows or 'gcore ' on Linux) to preserve any in-memory session tokens or credentials that a prior XSS or CSRF exploit may have harvested; (2) export the pgAdmin saved server definitions file ('servers.json' or the pgAdmin SQLite storage database at '%APPDATA%\pgAdmin 4\pgadmin4.db' / '~/.pgadmin/pgadmin4.db') before the upgrade overwrites it, as it contains connection credentials that may have been read by a SQLi exploit; (3) document all PostgreSQL roles and their

last-modified timestamps via 'SELECT usename, passwd, valuntil FROM pg_shadow;' before rotation to establish the pre-incident credential baseline.

Step 4: Recovery — After upgrading, verify the pgAdmin version reported in the application matches 9.16 or later. Re-enable access for legitimate users and confirm CSRF protections are active by reviewing application behavior. Monitor PostgreSQL database audit logs for any unauthorized write operations, schema changes, or anomalous queries that may have occurred during the exposure window, consistent with AU-6 and AU-11 (Audit Record Retention).

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting), NIST AU-11 (Audit Record Retention), NIST AC-17 (Remote Access), CIS 6.2 (Establish an Access Revoking Process), CIS 6.1 (Establish an Access Granting Process)

Compensating: Verify pgAdmin version post-upgrade: 'pip show pgadmin4 | grep Version' or inspect the pgAdmin About page. Confirm CSRF token enforcement is active by using curl to submit a POST to the pgAdmin login endpoint without a CSRF token and verifying a 400/403 rejection: 'curl -X POST http://localhost:5050/login -d "email=test&password=test" -v' — a missing CSRF token should result in rejection, not a redirect. For PostgreSQL audit log review, query pg_log for DDL events during the exposure window: 'grep -E "(CREATE|DROP|ALTER|INSERT|UPDATE|DELETE)" /var/log/postgresql/postgresql-*.log | awk -v start="" -v end="" \'$1>=start && $1<=end\''.

Evidence: Recovery re-enables user access, which will generate new session and authentication events that could obscure pre-incident activity. Before re-enabling access, snapshot the current PostgreSQL audit logs and pgAdmin session database in their post-eradication but pre-recovery state as a forensic baseline. Specifically preserve: the PostgreSQL 'pg_stat_activity' view output ('SELECT pid, usename, application_name, client_addr, query, state, query_start FROM pg_stat_activity;') and any pgAdmin-initiated queries still visible in 'pg_stat_statements' to distinguish attacker-issued queries from legitimate administrative activity during the exposure window.

Step 5: Post-Incident — Conduct a control gap review against NIST SI-4 (System Monitoring) and CIS 7.1 (Establish and Maintain a Vulnerability Management Process) to assess whether pgAdmin 4 was included in your asset inventory and patch cadence. Evaluate whether AI-integrated features in administrative tools receive security review before deployment. Document lessons learned regarding administrative tool exposure and access restriction policies (NIST AC-17, AC-6).

NIST Phase: Post Incident

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

Controls: NIST AC-6 (Least Privilege), NIST AC-17 (Remote Access), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory)

Compensating: Audit whether pgAdmin 4 appeared in your software inventory by running 'pip list | grep pgadmin' across managed hosts via a script or configuration management tool (Ansible ad-hoc: 'ansible all -m shell -a "pip show pgadmin4 2>/dev/null"'). Document any pgAdmin deployments discovered during this incident that were absent from CIS 1.1 inventory as a gap finding. For AI Assistant feature governance, establish a checklist item in your change management process requiring security review of any LLM-integrated or AI-assisted feature in privileged administrative tooling before enabling in production — specifically reviewing whether the feature can initiate write operations against backend data stores without explicit user confirmation.

Evidence: Post-incident activity does not alter live system state; however, preserve the complete incident timeline documentation before lessons-learned meetings, including: (1) the original pgAdmin version confirmed on each affected host at discovery; (2) firewall and WAF rule change logs from the containment phase; (3) the PostgreSQL audit log extracts covering the full exposure window (from last known-good patch date through eradication completion); (4) any pgAdmin session database snapshots taken during eradication showing saved server credentials potentially exposed via CVE-2026-12045 (SQLi) or CVE-2026-12048 (stored XSS). These artifacts support both the lessons-learned report and any regulatory breach notification assessment.

Detection Guidance

Focus detection on four behavioral patterns corresponding to each CVE. For CVE-2026-12048 (stored XSS): inspect pgAdmin application logs and browser-side CSP violation reports for injected script tags or encoded JavaScript payloads in object names, comments, or user-controlled fields. For CVE-2026-12046 (CSRF): look for POST requests to pgAdmin endpoints lacking valid CSRF token headers or originating from unexpected referrers; web proxy and WAF logs are the primary source. For CVE-2026-12045 (AI Assistant bypass): query PostgreSQL server logs for unexpected write, update, insert, or DDL operations attributed to sessions that should be operating in read-only mode, particularly those initiated through the AI Assistant interface. For CVE-2026-12044 (SQLi): monitor PostgreSQL logs for malformed queries, syntax errors, stacked queries, or time-delay patterns indicative of injection; enable pg_stat_statements if not active. Cross-reference with SI-7 (Software, Firmware, and Information Integrity) for configuration tampering and AC-2 (Account Management) for privilege escalation following session compromise. No confirmed public IOC signatures are available at this time; detection relies on behavioral and log-based indicators. Mapped controls: NIST AU-2, AU-6, AU-12; CIS 8.2.

Framework Mappings

MITRE-ATTACK

- **T1190** — Exploit Public-Facing Application
- **T1185** — Browser Session Hijacking
- **T1565.001** — Stored Data Manipulation
- **T1059.007** — JavaScript

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
- **SI-10** — Information Input Validation
- **AC-3** — Access Enforcement
- **SC-23** — Session Authenticity

OWASP-TOP10-2021

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

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures
- **6.1** — Establish an Access Granting Process

- **6.8** — Define and Maintain Role-Based Access Control

ISO-27001-2022

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1190	Exploit Public-Facing Application	Initial-Access
T1185	Browser Session Hijacking	Collection
T1565.001	Stored Data Manipulation	Impact
T1059.007	JavaScript	Execution

Sources

Source	URL	Tier
CVE-2026-12048 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-12048	T1
Vulnerability Intelligence Report — June 20, 2026	https://threat-modeling.com/vulnerability-intelligence-report-june-...	T3
pgAdmin4 < 9.16 Multiple Vulnerabilities | Tenable®	https://www.tenable.com/plugins/pipeline/issues/222825	T3
CVE-2026-45445 - Red Hat Customer Portal	https://access.redhat.com/security/cve/cve-2026-45445	T3
May 2026 CVE Landscape - Recorded Future	https://www.recordedfuture.com/blog/may-2026-cve-landscape	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-12048, CVE-2026-12046, CV...	T1

Generated 2026-06-22 06:23 UTC by TJS Security Command Center