INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-21 19:03 UTC

PgBouncer Integer Overflow in SCRAM Packet Parsing Enables Unauthenticated Remote Crash (CVE-2026-6664)

CVE VULNERABILITY | **HIGH** | CVSS 7.5 | **CISA KEV**

SCC Item ID	SCC-CVE-2026-0210
Type	CVE Vulnerability
CVE ID	CVE-2026-6664
Severity	HIGH
CVSS Base Score	7.5
EPSS Score	0.0005 (14th percentile)
KEV Status	Yes — CISA Known Exploited Vulnerability
Affected Products	PgBouncer before 1.25.2
Published	2026-05-20T00:00:00Z
Discovery Source	Vulncheck Kev

Executive Summary

A confirmed, actively exploited vulnerability in PgBouncer, a widely deployed PostgreSQL connection pooler, allows an unauthenticated attacker to crash the service by sending a single malformed network packet. Any organization running PgBouncer versions prior to 1.25.2 with database infrastructure exposed to untrusted networks is at risk of database connection disruption and application downtime. The vulnerability appears in both the CISA KEV and VulnCheck KEV catalogs, indicating real-world exploitation and mandating immediate remediation.

Technical Analysis

CVE-2026-6664 is an integer overflow (CWE-190) in PgBouncer's SCRAM authentication packet parsing logic. An unauthenticated remote attacker can send a crafted SCRAM packet that triggers the overflow, bypassing a boundary check and causing a denial-of-service crash. No authentication is required; the attack surface is the PgBouncer listening port (default TCP 6432). Affected versions: PgBouncer < 1.25.2. Fixed version: PgBouncer 1.25.2. CVSS base score: 7.5 (High). MITRE ATT&CK technique: T1499.004 (Endpoint Denial of Service: Application or System Exploitation). The vulnerability is confirmed actively exploited per CISA KEV and VulnCheck KEV. No CVSSv3 vector string was provided by the vendor at time of publication. EPSS score of

0.00046 (14th percentile) reflects low automated scanning activity, but active exploitation status supersedes that signal.

Action Checklist

1. Step 1: Containment, Identify all PgBouncer instances running versions prior to 1.25.2 using your asset inventory (CIS 1.1). Immediately restrict network access to the PgBouncer port (default TCP 6432) to trusted application servers only, using host-based firewall rules (CIS 4.4) or network ACLs. Do not leave PgBouncer exposed to untrusted or internet-facing networks without access controls.
2. Step 2: Detection, Query logs and SIEM for anomalous connection patterns to the PgBouncer port: high-volume unauthenticated connections, repeated SCRAM handshake failures, and sudden process crashes or restarts. Review PgBouncer log files (typically /var/log/pgbouncer/pgbouncer.log) for 'FATAL', 'ERROR', or unexpected termination entries. Alert on repeated failed SCRAM authentication attempts from external or unexpected source IPs (NIST AU-6, AU-2). Enable process restart monitoring for the pgbouncer service via your host-based monitoring agent.
3. Step 3: Eradication, Upgrade all PgBouncer instances to version 1.25.2 or later. Obtain the release from the official PgBouncer GitHub repository (https://github.com/pgbouncer/pgbouncer/releases) or your OS package manager. Validate the installed version with 'pgbouncer --version' post-upgrade. For Linux distributions, apply vendor-issued packages when available (Debian, Ubuntu, RHEL). Reference Tenable plugin 313599 and CISA KEV for remediation confirmation criteria (NIST SI-1, CIS 7.3, CIS 7.4).
4. Step 4: Recovery, After patching, restart PgBouncer and confirm the service starts cleanly with no crash loops. Validate that application database connections resume normally. Monitor the PgBouncer process for 24-48 hours post-patch for stability. Re-enable any temporarily restricted network access only after version 1.25.2 is confirmed running. Document the remediation timestamp for audit and compliance records (NIST IR-5, AU-11).
5. Step 5: Post-Incident, Review whether PgBouncer (and other middleware services) are included in your vulnerability scanning scope and patch management program (CIS 7.1, CIS 7.2). Assess whether SCRAM-capable services are unnecessarily exposed to untrusted networks and enforce least-privilege network segmentation. Verify your incident response plan (NIST IR-8) covers database middleware and connection pooling services, not just application servers. Add PgBouncer to automated vulnerability scan coverage if not already present.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to senior IR leadership and notify database/application owners immediately if PgBouncer crash events are observed in production logs prior to patching, if the PgBouncer port (TCP 6432) is confirmed reachable from untrusted or internet-facing networks, or if application downtime resulting from exploit-triggered crashes affects systems processing PII, PHI, or financial data subject to breach notification obligations.

Recovery Notes	After upgrading to PgBouncer 1.25.2, confirm that SCRAM authentication handshakes complete successfully end-to-end by running authenticated test queries through the connection pooler before restoring unrestricted application access. Monitor /var/log/pgbouncer/pgbouncer.log and systemd journal for the pgbouncer service continuously for 24-48 hours post-patch, specifically watching for any recurrence of FATAL or unexpected termination entries that could indicate a misconfiguration introduced during upgrade. Retain firewall restrictions on TCP 6432 to trusted application server CIDRs as a permanent network segmentation control, not just a temporary compensating measure.
Forensic Artifacts	PgBouncer application log (/var/log/pgbouncer/pgbouncer.log): Contains timestamped FATAL/ERROR entries and unexpected termination events caused by the integer overflow in SCRAM packet parsing — the exploit sends a single malformed SCRAM packet that crashes the process, so look for abrupt log termination followed by service restart entries. | Systemd journal for the pgbouncer service ('journalctl -u pgbouncer'): Records service start/stop/crash cycles; a CVE-2026-6664 exploitation attempt would appear as repeated unexpected 'Stopped' and 'Started' entries in tight temporal clusters corresponding to crash-restart loops triggered by malformed SCRAM packets. | Process coredump files (/var/lib/systemd/coredump/ or /tmp/core.*): An integer overflow triggering a process crash on malformed SCRAM packet input may produce a core dump from the pgbouncer process, which can be analyzed with 'gdb pgbouncer ' to confirm the faulting instruction and call stack within the SCRAM parsing code path. | Network packet capture on TCP 6432: A Wireshark/tcpdump capture ('tcpdump -i -w /ir/evidence/pgbouncer_traffic.pcap tcp port 6432') taken during or immediately after suspected exploitation will preserve the malformed SCRAM client-first-message packet — the integer overflow is triggered by a crafted packet with an oversized or malformed length field in the SCRAM handshake, which is identifiable in the raw packet payload. | Host firewall and connection logs (iptables LOG target or /var/log/ufw.log on Ubuntu): Pre-containment connection records to TCP 6432 identify attacker source IPs that sent unauthenticated SCRAM packets, allowing correlation with threat intelligence feeds and CISA KEV attribution data for CVE-2026-6664 exploitation campaigns.

Per-Action IR Details

Step 1: Containment — Identify all PgBouncer instances running versions prior to 1.25.2 using your asset inventory (CIS 1.1). Immediately restrict network access to the PgBouncer port (default TCP 6432) to trusted application servers only, using host-based firewall rules (CIS 4.4) or network ACLs. Do not leave PgBouncer exposed to untrusted or internet-facing networks without access controls.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST SC-7 (Boundary Protection), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: Run 'ss -tlnp | grep 6432' or 'netstat -tlnp | grep 6432' on each host to confirm PgBouncer listener exposure. Apply an immediate iptables rule to restrict access: 'iptables -I INPUT -p tcp --dport 6432 ! -s -j DROP'. For inventory discovery across subnets, use 'nmap -p 6432 --open /24' to identify all exposed PgBouncer instances before patching. Document each discovered instance with its IP, version, and firewall status.

Evidence: Before restricting access, capture current iptables/nftables rules ('iptables-save > pre_containment_rules.txt'), active TCP connections to port 6432 ('ss -tnp sport = :6432 > active_connections.txt'), and a list of source IPs that have connected to the PgBouncer port in the past 24-72 hours from /var/log/pgbouncer/pgbouncer.log. Capture PgBouncer process state with 'ps auxf | grep pgbouncer' and note the running binary path to confirm version pre-patch.

Step 2: Detection — Query logs and SIEM for anomalous connection patterns to the PgBouncer port: high-volume unauthenticated connections, repeated SCRAM handshake failures, and sudden process crashes or restarts. Review PgBouncer log files (typically /var/log/pgbouncer/pgbouncer.log) for 'FATAL', 'ERROR', or unexpected termination entries. Alert on repeated failed SCRAM authentication attempts from external or unexpected source IPs (NIST AU-6, AU-2). Enable process restart monitoring for the pgbouncer service via your host-based monitoring agent.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis

Controls: NIST AU-2 (Event Logging), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Parse PgBouncer logs directly for crash indicators: 'grep -E "FATAL|ERROR|terminated|signal|core dumped" /var/log/pgbouncer/pgbouncer.log | tail -200'. Detect SCRAM-specific failure spikes: 'grep -c "SCRAM" /var/log/pgbouncer/pgbouncer.log' and 'grep "SCRAM" /var/log/pgbouncer/pgbouncer.log | awk "{print $1, $2, $NF}" | sort | uniq -c | sort -rn | head -20'. Monitor service restarts without SIEM using: 'journalctl -u pgbouncer --since "24 hours ago" | grep -E "Started|Stopped|Failed"'. Use a Sigma rule targeting pgbouncer process crash events in systemd journal logs for teams with log shippers (e.g., Filebeat + Elasticsearch).

Evidence: Capture the full PgBouncer log file prior to any log rotation: 'cp /var/log/pgbouncer/pgbouncer.log /ir/evidence/pgbouncer_pre_patch_$(date +%Y%m%d%H%M%S).log'. Collect systemd journal entries for the pgbouncer service: 'journalctl -u pgbouncer --since "72 hours ago" > /ir/evidence/pgbouncer_journal.txt'. Record any coredump files in /var/lib/systemd/coredump/ or /tmp that correspond to the pgbouncer process — a successful CVE-2026-6664 exploit via malformed SCRAM packet would produce an abnormal process termination potentially leaving a core dump. Capture kernel syslog entries: 'grep pgbouncer /var/log/syslog > /ir/evidence/pgbouncer_syslog.txt'.

Step 3: Eradication — Upgrade all PgBouncer instances to version 1.25.2 or later. Obtain the release from the official PgBouncer GitHub repository (https://github.com/pgbouncer/pgbouncer/releases) or your OS package manager. Validate the installed version with 'pgbouncer --version' post-upgrade. For Linux distributions, apply vendor-issued packages when available (Debian, Ubuntu, RHEL). Reference Tenable plugin 313599 and CISA KEV for remediation confirmation criteria (NIST SI-1, CIS 7.3, CIS 7.4).

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)

Compensating: For Debian/Ubuntu: 'apt-get update && apt-get install --only-upgrade pgbouncer && pgbouncer --version'. For RHEL/Rocky/AlmaLinux: 'dnf update pgbouncer && pgbouncer --version'. If building from source (no distro package for 1.25.2 yet), download the tarball from the official GitHub releases page, verify the SHA256 checksum published in the release notes before installation, then compile and install. Post-upgrade, verify: 'pgbouncer --version | grep -E "1\.2[5-9]|1\.[3-9]"' to confirm the vulnerable version is no longer running. Run Tenable Nessus (free for personal use) with plugin 313599 or use OpenVAS to confirm the finding is resolved.

Evidence: Before upgrading, capture the binary hash of the vulnerable PgBouncer executable: 'sha256sum $(which pgbouncer) > /ir/evidence/pgbouncer_vuln_binary_hash.txt' and record the running version: 'pgbouncer --version >> /ir/evidence/pgbouncer_vuln_binary_hash.txt'. Preserve the pgbouncer.ini configuration file: 'cp /etc/pgbouncer/pgbouncer.ini /ir/evidence/pgbouncer_ini_pre_patch.bak'. These preserve the pre-patch state for audit and confirm the vulnerable version was present, which is required for CISA KEV reporting and internal incident documentation.

Step 4: Recovery — After patching, restart PgBouncer and confirm the service starts cleanly with no crash loops. Validate that application database connections resume normally. Monitor the PgBouncer process for 24-48 hours post-patch for stability. Re-enable any temporarily restricted network access only after version 1.25.2 is confirmed running. Document the remediation timestamp for audit and compliance records (NIST

IR-5, AU-11).

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-5 (Incident Monitoring), NIST AU-11 (Audit Record Retention), NIST CP-10 (System Recovery and Reconstitution), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: Confirm clean service start: 'systemctl restart pgbouncer && systemctl status pgbouncer' — look for 'active (running)' with no 'failed' units. Validate SCRAM authentication is completing successfully post-patch by running a test connection: 'psql -h -p 6432 -U -d -c "SELECT 1;"' and confirming it returns without error. Monitor for crash loops: 'watch -n 5 systemctl is-active pgbouncer' for the first 30 minutes post-restart. Re-enable restricted firewall rules only after 'pgbouncer --version' confirms 1.25.2 or later is active. Log the exact timestamp of patching and service restoration to a change record for AU-11 compliance.

Evidence: Capture post-patch version confirmation as a timestamped artifact: 'pgbouncer --version && date > /ir/evidence/pgbouncer_post_patch_version.txt'. Record the first successful authenticated connection after recovery in PgBouncer logs to establish a clean-state baseline. Preserve the updated firewall rules: 'iptables-save > /ir/evidence/post_recovery_iptables.txt'. Retain these artifacts per your AU-11 audit record retention schedule — they document the remediation window for any compliance review tied to CISA KEV obligations.

Step 5: Post-Incident — Review whether PgBouncer (and other middleware services) are included in your vulnerability scanning scope and patch management program (CIS 7.1, CIS 7.2). Assess whether SCRAM-capable services are unnecessarily exposed to untrusted networks and enforce least-privilege network segmentation. Verify your incident response plan (NIST IR-8) covers database middleware and connection pooling services, not just application servers. Add PgBouncer to automated vulnerability scan coverage if not already present.

NIST Phase: Post Incident

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

Controls: 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: Use OpenVAS or Trivy (for containerized PgBouncer deployments) to add PgBouncer to recurring scan scope and confirm coverage. Enumerate all middleware services listening on non-standard ports with: 'ss -tlnp | grep -v ":22\|:80\|:443"' and cross-reference against your asset inventory (CIS 1.1) to identify other unscanned services in the same class as PgBouncer. Document a lessons-learned entry specifically noting that SCRAM authentication protocol handling in connection poolers represents a crash-via-unauthenticated-packet attack surface, and update IR runbooks to include middleware service recovery procedures. Schedule a tabletop exercise or checklist review for database middleware within 30 days.

Evidence: Produce a post-incident timeline document that records: first observed crash or anomaly timestamp (from pgbouncer.log), containment action timestamp (iptables rule application), patch application timestamp, and recovery confirmation timestamp. Retain the pre- and post-patch binary hashes, the preserved pgbouncer.ini, and the journal logs collected during detection as the evidentiary record for this incident. If the organization is subject to regulatory requirements (e.g., PCI-DSS for database environments), retain these artifacts for the required audit period per AU-11 policy.

Detection Guidance

Primary detection signal: unexpected PgBouncer process crashes or rapid restart cycles. Check the PgBouncer log (default: /var/log/pgbouncer/pgbouncer.log or journald unit 'pgbouncer') for FATAL errors, segfaults, or 'server login failed' entries clustering from a single source IP. In your SIEM, alert on: (1) repeated failed SCRAM authentication events from external IPs within a short time window, (2) the pgbouncer process exiting with non-zero status codes, and (3) spikes in TCP SYN connections to port 6432 from unexpected sources. MITRE

ATT&CK T1499.004 (Application Denial of Service) is the relevant technique; hunting queries should focus on availability disruption patterns rather than data exfiltration indicators. No specific IOCs (IPs, hashes, domains) are confirmed publicly at this time. Prioritize detection of service disruption over network-layer indicators. Reference NIST AU-6 and AU-2 for log review and alerting requirements.

Framework Mappings

MITRE-ATTACK

- **T1499.004** — Application or System Exploitation

NIST-800-53R5

- **SI-16** — Memory Protection

CIS-V8

- **16.10** — Apply Secure Design Principles in Application Architectures

ISO-27001-2022

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

NIST-CSF-2

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1499.004	Application or System Exploitation	Impact

Sources

Source	URL	Tier
vulncheck_kev	https://nvd.nist.gov/vuln/detail/CVE-2026-6664	T1
CVE-2026-6665: PgBouncer Buffer Overflow Vulnerability	https://www.sentinelone.com/vulnerability-database/cve-2026-6665/	T3
DEBIAN-CVE-2026-6664 | Mondoo Vulnerability Intelligence	https://mondoo.com/vulnerability-intelligence/vulnerability/DEBIAN-...	T3
Linux Distros Unpatched Vulnerability : CVE-2026-6664 | Tenable®	https://www.tenable.com/plugins/nessus/313599	T3

Source	URL	Tier
Known Exploited Vulnerabilities Catalog | CISA	https://www.cisa.gov/known-exploited-vulnerabilities-catalog	**T1**

Generated 2026-05-21 19:03 UTC by TJS Security Command Center