INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-10 07:27 UTC

CVE-2026-11457: A security flaw has been discovered in erzhongxmu JeeWMS up to 141740afb2ba14d441c82a833d0a418d07ca2...

CVE VULNERABILITY | HIGH | CVSS 7.3

SCC Item ID	SCC-CVE-2026-0288
Type	CVE Vulnerability
CVE ID	CVE-2026-11457
Severity	HIGH
CVSS Base Score	7.3
EPSS Score	0.0005 (15th percentile)
Affected Products	erzhongxmu JeeWMS up to commit 141740afb2ba14d441c82a833d0a418d07ca2d69 (rolling release, no fixed version confirmed)
Published	2026-06-07T09:16:21.843
Discovery Source	Nvd

Executive Summary

A publicly exploitable injection vulnerability (CVE-2026-11457, CVSS 7.3) has been identified in JeeWMS, an open-source warehouse management system maintained by erzhongxmu. The flaw targets a database test-connection endpoint and allows remote attackers to manipulate database parameters without authentication. Organizations running JeeWMS in internet-accessible environments face direct risk of data exposure or backend compromise; no patched version exists as of this writing.

Technical Analysis

CVE-2026-11457 affects erzhongxmu JeeWMS through commit 141740afb2ba14d441c82a833d0a418d07ca2d69 (rolling release). The vulnerable endpoint is /base-boot/jmreport/testConnection within the JimuReport test-connection component. Attackers can manipulate the dbType, dbDriver, dbUrl, dbUsername, and/or dbPassword parameters to trigger an injection condition (CWE-74: Injection; CWE-707: Improper Neutralization). Exploitation is unauthenticated and network-accessible. This maps to MITRE ATT&CK T1190 (Exploit Public-Facing Application). A public exploit has been released. CVSS base score is 7.3 (high). EPSS score is 0.00047 (14.8th percentile), indicating currently low observed exploitation activity. No vendor patch exists; the vendor did not respond to pre-disclosure notification. Source: NVD (https://nvd.nist.gov/vuln/detail/CVE-2026-11457).

Action Checklist

1. Step 1: Containment. Immediately block external access to the /base-boot/jmreport/testConnection endpoint via WAF rule or network-layer ACL. Verify no internet-facing JeeWMS instance exposes this path. Restrict internal access to trusted IP ranges only (NIST AC-4: Information Flow Enforcement; CIS 4.4: Implement and Manage a Firewall on Servers).
2. Step 2: Detection. Search web server and application logs for POST or GET requests to /base-boot/jmreport/testConnection. Flag requests containing unusual dbUrl values (e.g., JDBC strings pointing to external hosts), unexpected dbDriver class names, or rapid sequential calls to this endpoint. Correlate with SIEM alerts for T1190 (Exploit Public-Facing Application). Enable detailed request logging at the application layer if not already active (NIST AU-2: Event Logging; CIS 8.2: Collect Audit Logs).
3. Step 3: Eradication. No vendor-supplied patch is available. Remove or disable the /base-boot/jmreport/testConnection endpoint at the application configuration level if JimuReport test-connection functionality is not required in production. If the feature is required, enforce strict server-side allowlisting of permitted dbDriver and dbUrl values. Monitor the erzhongxmu repository for commits addressing this endpoint (CIS 7.1: Establish and Maintain a Vulnerability Management Process).
4. Step 4: Recovery. After blocking the endpoint, validate that no unauthorized database connections were established by reviewing database access logs for connections originating from the JeeWMS application server during the exposure window. Rotate database credentials used by JeeWMS (MITRE D3FEND D3-CRO: Credential Rotation). Confirm WAF/ACL rules are active and logging. Establish ongoing monitoring for requests targeting this path (NIST AU-6: Audit Record Review, Analysis, and Reporting).
5. Step 5: Post-Incident. Assess whether internal processes require vendor responsiveness as a procurement criterion for open-source dependencies. Review all JimuReport-integrated endpoints for similar unauthenticated parameter-handling patterns. Document this exposure as evidence for control gap review under least-privilege access to database configuration interfaces (NIST AC-6: Least Privilege; CIS 7.2: Establish and Maintain a Remediation Process).

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to senior IR lead and notify data owners immediately if database access logs confirm any successful unauthorized JDBC connection originating from the JeeWMS host during the exposure window, if the JeeWMS database contains PII or inventory data subject to breach notification requirements, or if the two-person team lacks the capability to rebuild or reconfigure the JeeWMS WAR to disable the endpoint within the containment window.
Recovery Notes	After credential rotation and endpoint blocking are confirmed, monitor the JeeWMS application server's outbound network connections for a minimum of 14 days for any JDBC or RMI connections to non-internal hosts, which would indicate a previously injected payload establishing persistence via the testConnection mechanism. Validate database integrity by comparing row counts and schema state against the most recent known-good backup taken before the exposure window. Maintain the `iptables` or nginx block rule for `/base-boot/jmreport/testConnection` indefinitely until erzhongxmu publishes a commit that removes or secures the endpoint and the fix is validated against the JeeWMS source.

Forensic Artifacts	Tomcat `localhost_access_log.*.txt` and `catalina.out`: Primary source for all HTTP requests to `/base-boot/jmreport/testConnection`, including source IP, timestamp, HTTP method, response code, and (if body logging is enabled) the raw `dbUrl`, `dbDriver`, `dbUsername`, and `dbPassword` parameter values submitted by an attacker. | MySQL/MariaDB general query log or MariaDB Audit Plugin log: Records all `Connect` and `Query` events at the database layer; cross-referencing connection timestamps and source IPs against the JeeWMS app server IP during the exposure window reveals whether injected JDBC parameters successfully redirected database connections to attacker-controlled hosts. | Network flow records or `iptables` connection tracking logs (via `conntrack -L` output saved during triage): Captures any outbound TCP connections from the JeeWMS server on ports 3306 (MySQL), 1099 (RMI), or 389/636 (LDAP) to external IPs, which would indicate JDBC-based SSRF or RMI deserialization attempts triggered through the testConnection endpoint. | JeeWMS application configuration files (`application.yml` / `application.properties`): Documents the datasource credentials and JDBC URL in use at the time of exposure; required to assess blast radius and confirm which database accounts must be rotated, and to verify whether the JeeWMS service account had excessive database privileges that amplified exploit impact. | WAF or reverse proxy (nginx/Apache) access and error logs: Secondary corroboration for endpoint access patterns; error log entries showing 500-class responses to `/base-boot/jmreport/testConnection` may indicate exploitation attempts that triggered backend exceptions during JDBC driver instantiation with malformed or malicious class names.

Per-Action IR Details

Step 1: Containment — Immediately block external access to the /base-boot/jmreport/testConnection endpoint via WAF rule or network-layer ACL. Verify no internet-facing JeeWMS instance exposes this path. Restrict internal access to trusted IP ranges only (NIST AC-4: Information Flow Enforcement; CIS 4.4: Implement and Manage a Firewall on Servers).

NIST Phase: Containment

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

Controls: NIST AC-4 (Information Flow Enforcement), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: On Linux-hosted JeeWMS: run `iptables -I INPUT -p tcp --dport 8080 -m string --string '/jmreport/testConnection' --algo bm -j DROP` to block at the kernel level without a WAF. On nginx reverse proxy: add `location ~ ^/base-boot/jmreport/testConnection { deny all; return 403; }` in the server block. Verify with `curl -X POST http://localhost:8080/base-boot/jmreport/testConnection` from a non-trusted IP — a 403 or connection refused confirms the block. Document the rule and timestamp for the incident timeline.

Evidence: Before applying the block rule, capture a snapshot of current active network connections to the JeeWMS application server using `ss -tnp | grep :8080` (or the configured port) and save output with timestamp. Dump the web server access log (e.g., `/var/log/nginx/access.log` or Tomcat `logs/localhost_access_log.*.txt`) to preserve any prior POST requests to `/base-boot/jmreport/testConnection` that occurred before containment. Preserve the raw log file with `md5sum` hash for evidentiary integrity per NIST 800-61r3 §3.2 evidence handling guidance.

Step 2: Detection — Search web server and application logs for POST or GET requests to /base-boot/jmreport/testConnection. Flag requests containing unusual dbUrl values (e.g., JDBC strings pointing to external hosts), unexpected dbDriver class names, or rapid sequential calls to this endpoint. Correlate with SIEM alerts for T1190 (Exploit Public-Facing Application). Enable detailed request logging at the application layer if not already active (NIST AU-2: Event Logging; CIS 8.2: Collect Audit Logs).

NIST Phase: Detection Analysis

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

Controls: NIST AU-2 (Event Logging), CIS 8.2 (Collect Audit Logs)

Compensating: Without a SIEM, run this grep against Tomcat or nginx access logs to surface all hits on the endpoint: `grep -E 'jmreport/testConnection' /var/log/nginx/access.log | awk '{print $1, $7, $9}' > testconn_hits.txt`. To extract POST body parameters containing JDBC strings, enable Tomcat request body logging via `AccessLogValve` with `pattern="%h %t %r %s %b %{postbody}r"` and restart. For JDBC exfil patterns, use: `grep -Ei '(jdbc:|rmi:|ldap:|http://)' testconn_hits.txt` to identify dbUrl values pointing to external or unusual hosts. Ship results to a central log file reviewed daily by the two-person team.

Evidence: Collect the full Tomcat `catalina.out` and `localhost_access_log.*.txt` files covering the entire period JeeWMS was internet-accessible. Extract all POST request bodies to `/base-boot/jmreport/testConnection` — specifically the `dbUrl`, `dbDriver`, `dbUsername`, and `dbPassword` parameter values, which an attacker would manipulate to redirect the JDBC connection. Preserve any WAF logs if present. Check for evidence of MITRE T1190 exploitation: sequential rapid requests (automated scanning) or single crafted requests with non-default JDBC driver class names (e.g., `com.mysql.jdbc.Driver` replaced with a malicious class loader path).

Step 3: Eradication — No vendor-supplied patch is available. Remove or disable the /jmreport/testConnection endpoint at the application configuration level if JimuReport test-connection functionality is not required in production. If the feature is required, enforce strict server-side allowlisting of permitted dbDriver and dbUrl values. Monitor the erzhongxmu repository for commits addressing this endpoint (CIS 7.1: Establish and Maintain a Vulnerability Management Process).

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Since no patch exists as of the CVE-2026-11457 disclosure, disable the JimuReport test-connection servlet directly: locate the `JimuReportTokenService` or equivalent route registration in `JeeWMS/jeewms-web/src/main/java` and comment out the `/testConnection` `@PostMapping` annotation, then rebuild the WAR. If a rebuild is not feasible, use a Tomcat `web.xml` security constraint to block the path: add `/base-boot/jmreport/testConnection`. Set a GitHub watch/RSS alert on `https://github.com/erzhongxmu/JeeWMS/commits/master` filtered to changes in files matching `*testConnection*` or `*jmreport*`.

Evidence: Before disabling the endpoint, document the exact JeeWMS commit hash currently deployed (visible in `git log --oneline -1` on the server or in the deployment manifest) to establish the vulnerable baseline for the incident record. Capture the current application configuration files under `src/main/resources/` and any `application.yml` or `application.properties` that define datasource parameters — these confirm what database credentials and connection strings were exposed to the vulnerable endpoint during the window.

Step 4: Recovery — After blocking the endpoint, validate that no unauthorized database connections were established by reviewing database access logs for connections originating from the JeeWMS application server during the exposure window. Rotate database credentials used by JeeWMS (MITRE D3FEND D3-CRO: Credential Rotation). Confirm WAF/ACL rules are active and logging. Establish ongoing monitoring for requests targeting this path (NIST AU-6: Audit Record Review, Analysis, and Reporting).

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST AU-6 (Audit Record Review, Analysis, And Reporting)

Compensating: Query the MySQL or MariaDB general query log (enable with `SET GLOBAL general_log = 'ON'; SET GLOBAL general_log_file='/var/log/mysql/general.log';`) and run: `grep -i 'Connect' /var/log/mysql/general.log | grep '"` to identify all connection events from the JeeWMS host during the exposure window. To rotate JeeWMS database credentials: `ALTER USER 'jeewms_user'@'localhost' IDENTIFIED BY '';` then update `application.yml` (`spring.datasource.password`) and restart the service. Set a cron job `0 * * * * grep 'jmreport/testConnection' /var/log/nginx/access.log >> /var/log/jeewms_monitor.log` to maintain ongoing endpoint visibility.

Evidence: Pull the MySQL/MariaDB `general_log` or `audit_log` (if MariaDB Audit Plugin is enabled) for the full exposure window — specifically filter for `Connect` and `Query` events originating from the JeeWMS application server IP that occurred outside normal application operating hours or that used the JeeWMS service account to execute

DDL/DML not consistent with normal WMS operations. Also capture the database error log (`/var/log/mysql/error.log`) for failed connection attempts using injected JDBC parameters, which would appear as authentication errors from unexpected source IPs or driver class instantiation failures.

Step 5: Post-Incident — Assess whether internal processes require vendor responsiveness as a procurement criterion for open-source dependencies. Review all JimuReport-integrated endpoints for similar unauthenticated parameter-handling patterns. Document this exposure as evidence for control gap review under least-privilege access to database configuration interfaces (NIST AC-6: Least Privilege; CIS 7.2: Establish and Maintain a Remediation Process).

NIST Phase: Post Incident

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

Controls: NIST AC-6 (Least Privilege), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Enumerate all JimuReport-derived endpoints in the deployed JeeWMS WAR by running: `jar -tf jeewms.war | grep -i 'jmreport'` and cross-reference against the JimuReport source on GitHub (github.com/jeecgboot/JimuReport) for any other unauthenticated controller methods accepting JDBC or driver parameters. Use `grep -rn '@AnonymousAccess\|permitAll\|noAuth' src/` on the JeeWMS source to identify all endpoints that bypass authentication. Document findings in a control gap register. For the vendor responsiveness criterion, create a checklist item in procurement review: confirm open-source dependencies have active maintainers with sub-30-day CVE response history before approval.

Evidence: Compile the complete incident timeline from log artifacts collected in Steps 1–4: first observed request to `/base-boot/jmreport/testConnection`, all unique source IPs, all JDBC parameter values submitted, and any database-side anomalies. This timeline serves as the post-incident evidence package. Also document the JeeWMS commit hash at time of exposure (captured in Step 3) as the definitive vulnerable version reference, since JeeWMS uses a rolling release with no version tags — the commit hash `141740afb2ba14d441c82a833d0a418d07ca2d69` is the authoritative boundary for vulnerability scope.

Detection Guidance

Query web/application server logs for requests to /base-boot/jmreport/testConnection. Flag: (1) any dbUrl values containing external hostnames or IP addresses not in your approved database server list; (2) dbDriver values referencing non-standard JDBC driver classes; (3) high-frequency calls to this endpoint from a single source IP within a short window. In a SIEM, build a detection rule matching HTTP requests to this URI path with a non-2xx response code or anomalous parameter length. Behavioral indicator: unexpected outbound database-protocol traffic (default ports 3306, 5432, 1433, 1521) from the JeeWMS application server host. No confirmed public IOCs (IPs, hashes, domains) are available at this time. Reference: NIST AU-2 (Event Logging), CIS 8.2 (Collect Audit Logs).

Framework Mappings

MITRE-ATTACK

- **T1190** — Exploit Public-Facing Application

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

OWASP-TOP10-2021

- **A03:2021** — Injection

CIS-V8

- **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.8.8** — Management of technical vulnerabilities
- **A.5.21** — Managing information security in the ICT supply chain

SOC2-TSC

- **CC9.2** — Manages risks associated with vendors and business partners

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1190	Exploit Public-Facing Application	Initial-Access

Sources

Source	URL	Tier
nvd	https://nvd.nist.gov/vuln/detail/CVE-2026-11457	T1
CVE-2026-11457 - Exploits & Severity - Feedly	https://feedly.com/cve/CVE-2026-11457	T3
CVE-2026-11457: Injection in erzhongxmu JeeWMS - Threat Radar	https://radar.offseq.com/threat/cve-2026-11457-injection-in-erzhong...	T3
CVE-2026-11457 - Enginsight Vulnerability Database	https://cve.enginsight.com/2026/11457/index.html	T3
CVE-2026-37457: FRRouting (FRR) DoS Vulnerability - SentinelOne	https://www.sentinelone.com/vulnerability-database/cve-2026-37457/	T3

DISCLAIMER

Generated 2026-06-10 07:27 UTC by TJS Security Command Center