INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-04-13 18:26 UTC

CVE-2026-5194: wolfSSL Cryptographic Bypass Puts 5 Billion Devices at Risk of Certificate Forgery

CVE VULNERABILITY | **CRITICAL** | CVSS 9.5

SCC Item ID	SCC-CVE-2026-0037
Type	CVE Vulnerability
CVE ID	CVE-2026-5194
Severity	CRITICAL
CVSS Base Score	9.5
EPSS Score	0.0003 (10th percentile)
Affected Products	wolfSSL versions prior to 5.9.1; embedded systems, IoT devices, industrial control systems (ICS), routers, automotive systems, and aerospace/military equipment utilizing wolfSSL for TLS
Published	2026-04-13T15:56:03
Discovery Source	Rss

Executive Summary

A critical cryptographic flaw in the wolfSSL TLS library (CVE-2026-5194, CVSS 9.5) allows attackers to forge digital certificates and bypass authentication, enabling man-in-the-middle attacks against systems that trust those certificates. wolfSSL is embedded across a broad set of devices in IoT, automotive, industrial control, and embedded ecosystems, making the attack surface exceptionally broad. A patch exists in wolfSSL 5.9.1 (released April 8, 2026), but firmware distribution chains, vendor update cycles, and SDK dependencies will delay remediation across a significant portion of the affected device landscape for weeks to months.

Technical Analysis

CVE-2026-5194 is a cryptographic signature verification bypass in wolfSSL prior to version 5.9.1. The flaw resides in ECDSA and related digital signature validation logic: an attacker can supply X.509 certificates containing undersized digests that wolfSSL incorrectly accepts as valid. This allows forged certificates to pass verification without a legitimate private key. Successful exploitation enables TLS man-in-the-middle interception (MITRE T1557, T1557.002), impersonation of trusted entities, and authentication bypass. Weakened cryptographic processing (T1600, T1600.001) and subversion of certificate trust chains (T1553.004) are the

primary technique mappings. Relevant CWEs: CWE-326 (Inadequate Encryption Strength), CWE-347 (Improper Verification of Cryptographic Signature), CWE-295 (Improper Certificate Validation). Affected scope: all wolfSSL releases prior to 5.9.1 deployed in IoT, embedded, ICS, automotive, router, aerospace, and military contexts. wolfSSL 5.9.1 was released April 8, 2026 with the fix. MariaDB is explicitly unaffected (uses OpenSSL). EPSS score is currently low at 0.035% (10th percentile), reflecting limited observed exploitation as of scoring date; this metric does not reduce urgency given the attack surface breadth. CVSS vendor score was not published as of this analysis. Note: NVD and wolfSSL advisory URLs are listed in sources but were not directly accessed for this output; treat them as required reference stops before operational decisions.

Action Checklist

1. Step 1: Containment, Immediately inventory all systems and firmware utilizing wolfSSL prior to 5.9.1. Prioritize internet-facing devices (routers, VPN endpoints, industrial control system gateways) and systems that perform TLS certificate validation as part of authentication workflows. Where patching is not yet possible, evaluate whether the affected service can be isolated from untrusted networks or placed behind a TLS-terminating proxy that performs independent certificate validation. Note: TLS-terminating proxy mitigation is most feasible for network-adjacent devices (routers, gateways); embedded devices performing direct TLS validation may not support this approach and should be prioritized for patching.
2. Step 2: Detection, Query asset management, SBOM repositories, and firmware bill-of-materials records for wolfSSL version strings (search for 'wolfSSL' or 'CyaSSL', the prior library name, in package manifests, firmware images, and dependency lists). Review TLS inspection logs and IDS/IPS alerts for anomalous certificate handshakes, particularly certificates presenting undersized or malformed digest fields. No public CVE-specific IOC signatures or IDS rules were available as of April 2026; monitor NVD, wolfSSL GitHub releases, and your IDS vendor for signatures released after this advisory publication date.
3. Step 3: Eradication, Upgrade wolfSSL to version 5.9.1 or later per the official wolfSSL release (released April 8, 2026). For devices where wolfSSL is bundled in vendor firmware, apply vendor-issued firmware updates as they become available; track vendor security advisories for your specific device manufacturers. For internal software builds that link wolfSSL as a dependency, rebuild and redeploy against 5.9.1.
4. Step 4: Recovery, After patching, validate that wolfSSL version 5.9.1 or later is confirmed active on all previously affected systems. Re-test TLS handshake behavior on critical services. Revoke and reissue any certificates that may have been in scope during a potential exploitation window if compromise is suspected. Resume normal network exposure for previously isolated devices only after patch confirmation.
5. Step 5: Post-Incident, Audit SBOM and firmware dependency tracking processes to determine how wolfSSL version exposure was discovered and how quickly. Evaluate gaps in third-party library visibility across embedded and IoT device inventories. Establish a recurring firmware dependency review cadence and confirm that downstream vendor patch notifications are covered by your vulnerability management program. Document devices where patching will remain delayed and define compensating controls and review dates for each.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO, legal, and potentially regulatory bodies immediately if any internet-facing device running wolfSSL prior to 5.9.1 shows evidence of a TLS handshake completing with a certificate whose signature cannot be verified by an independent validator (e.g., testssl.sh or OpenSSL), or if any ICS, automotive, or medical IoT device in scope is confirmed unpatched and internet-reachable, as these conditions meet active critical-infrastructure risk thresholds and may trigger sector-specific breach notification obligations (NERC CIP, FDA cybersecurity, or FedRAMP incident reporting).
Recovery Notes	After patching to wolfSSL 5.9.1, monitor all previously affected services for a minimum of 30 days using TLS certificate fingerprint baselining — any certificate fingerprint change not corresponding to a planned reissuance must be treated as a potential post-exploitation artifact of CVE-2026-5194 forgery. Pay particular attention to ICS gateways and VPN endpoints where a forged certificate accepted during the exploitation window may have enabled a persistent MITM session that survives the library patch if the attacker's position in the network path is not also removed. Verify that any internal or partner systems that established TLS sessions with affected devices during the potential exploitation window have flushed certificate caches and re-validated peer certificates against a trusted, independently-maintained CA bundle.
Forensic Artifacts	TLS handshake PCAP captures from wolfSSL-linked service ports (443, 8883/MQTT, 5684/CoAP, 4433) showing Certificate handshake messages (record type 0x16, handshake type 0x0B) with signature digest fields inconsistent with the claimed issuing CA — the specific forensic signature of CVE-2026-5194 certificate forgery exploitation. | Zeek ssl.log and x509.log entries with mismatched `validation_status` and `cert_chain_fuids` fields, where a handshake completed successfully despite an x509 validation failure — indicative of wolfSSL accepting a forged certificate that a correctly-implemented TLS stack would reject. | Pre- and post-patch firmware binary hash manifests (`sha256sum` output) for each affected embedded device, used to confirm no secondary firmware tampering occurred during the exploitation window and to establish chain of custody for any ICS or automotive device requiring regulatory evidence. | SBOM and firmware bill-of-materials records identifying wolfSSL or CyaSSL version strings below 5.9.1, extracted via Syft or manual `strings` grep, documenting the scope of exposure at time of discovery for incident timeline and regulatory reporting purposes. | Certificate fingerprint snapshots (`openssl s_client` output saved to file) collected from each affected TLS endpoint at three points — pre-containment, post-patch, and 30-day post-recovery — to detect any certificate substitution event attributable to a forged certificate being accepted by the vulnerable wolfSSL library during the exploitation window.

Per-Action IR Details

Step 1: Containment — Immediately inventory all systems and firmware utilizing wolfSSL prior to 5.9.1. Prioritize internet-facing devices (routers, VPN endpoints, industrial control system gateways) and systems that perform TLS certificate validation as part of authentication workflows. Where patching is not yet possible, evaluate whether the affected service can be isolated from untrusted networks or placed behind a TLS-terminating proxy that performs independent certificate validation.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST SC-17 (Public Key Infrastructure Certificates), NIST SC-23 (Session Authenticity), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 4.5 (Implement and Manage a Firewall on End-User Devices)

Compensating: For a 2-person team without enterprise asset management: run `grep -r 'wolfSSL\|CyaSSL' /usr/lib/ /usr/local/lib/ /opt/ /firmware/` on Linux-based devices and `Get-ChildItem -Recurse -Filter '*.so' | Select-String 'wolfSSL'` on accessible Windows hosts. For network isolation without a commercial proxy, deploy an nginx or

HAProxy instance configured for TLS termination with `ssl_verify_client on` and a pinned CA bundle as a forward-facing proxy in front of each vulnerable device; this offloads certificate validation to a non-wolfSSL stack. Use iptables or Windows Firewall to restrict inbound TLS (port 443, 8883 for MQTT, 5684 for CoAP) to the proxy's IP only for each unpatched device.

Evidence: Before isolating any device, capture the current TLS certificate chain presented by the affected service using `openssl s_client -connect : -showcerts 2>/dev/null | openssl x509 -noout -text` and save the full output — this establishes the pre-containment certificate baseline for later comparison if a forged certificate was already in use. Also capture `ss -tlnp` or `netstat -tlnp` output on each device to document active TLS listeners linked to wolfSSL processes, and collect firmware version strings via the device management interface or by running `strings | grep -i 'wolfSSL\|CyaSSL\|5\.[0-9]'` to confirm the exact library version before isolation.

Step 2: Detection — Query asset management, SBOM repositories, and firmware bill-of-materials records for wolfSSL version strings (search for 'wolfSSL' or 'CyaSSL' — the prior library name — in package manifests, firmware images, and dependency lists). Review TLS inspection logs and IDS/IPS alerts for anomalous certificate handshakes, particularly certificates presenting undersized or malformed digest fields. No public CVE-specific IOC signatures were available as of this analysis; monitor NVD, wolfSSL's GitHub, and your IDS vendor for rule updates referencing CVE-2026-5194.

NIST Phase: Detection Analysis

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

Controls: NIST SI-4 (System Monitoring), NIST SI-5 (Security Alerts, Advisories, and Directives), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 8.2 (Collect Audit Logs)

Compensating: Without a SIEM, perform SBOM scanning using Syft (`syft -o cyclonedx-json | grep -i wolf`) or Grype (`grype sbom:./sbom.json`) to enumerate wolfSSL/CyaSSL presence in firmware images and container layers. For network-level detection of anomalous TLS handshakes, capture traffic on the device's network interface using `tcpdump -i eth0 -w /tmp/tls_capture.pcap 'tcp port 443 or tcp port 8883'` and analyze in Wireshark with the display filter `tls.handshake.type == 11` (Certificate) to inspect presented certificate digest fields for truncation or malformation consistent with signature bypass. Deploy the Zeek `ssl` and `x509` log scripts if Zeek is available; look for `certificate.key_alg` anomalies or `validation_status != ok` paired with a successful handshake completion, which would indicate the library accepted an invalid certificate.

Evidence: Capture full TLS handshake PCAP files from all internet-facing wolfSSL-linked services before any rule-based filtering is applied — the specific artifact of CVE-2026-5194 exploitation would appear in the Certificate handshake message (TLS record type 0x16, handshake type 0x0B) as a signature digest field that does not correctly correspond to the claimed signing CA's key. Extract and archive Zeek `ssl.log` and `x509.log` entries, specifically the `cert_chain_fuids`, `validation_status`, and `subject` fields. Also collect any existing IDS/IPS alert logs from Snort or Suricata referencing TLS anomalies in the exploitation window, noting that pre-5.9.1 wolfSSL may complete a handshake flagged as invalid by a correct implementation on the opposing end.

Step 3: Eradication — Upgrade wolfSSL to version 5.9.1 or later per the official wolfSSL release (released April 8, 2026). For devices where wolfSSL is bundled in vendor firmware, apply vendor-issued firmware updates as they become available; track vendor security advisories for your specific device manufacturers. For internal software builds that link wolfSSL as a dependency, rebuild and redeploy against 5.9.1.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST SI-7 (Software, Firmware, and Information Integrity), NIST CM-3 (Configuration Change Control), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 2.2 (Ensure Authorized Software is Currently Supported)

Compensating: For internal builds, verify the linked wolfSSL version post-rebuild by running `strings | grep -E 'wolfSSL [0-9]+\.[0-9]+\.[0-9]+'` against the recompiled binary before redeployment to confirm 5.9.1 is linked and not a cached older version from a build artifact. For vendor-managed firmware where an update is unavailable, subscribe to the specific device manufacturer's security advisory RSS feed or mailing list and set a calendar-based 72-hour review

cadence; document each unpatched device in a compensating control register with justification, per NIST SI-2 requirements. Use AIDE or Tripwire (open-source) to baseline firmware file hashes post-patch so any unauthorized firmware rollback is detectable.

Evidence: Before executing the wolfSSL 5.9.1 upgrade on each system, preserve a forensic image or hash manifest of the existing firmware or binary containing the vulnerable wolfSSL library — use `sha256sum ` or `dd if=/dev/mtd0 | sha256sum` for embedded flash — to establish a pre-patch chain of custody artifact. This is critical for ICS and automotive systems where the vulnerable library may have been exploited prior to patch; an unchanged hash post-incident confirms no secondary tampering occurred to the firmware during an exploitation window.

Step 4: Recovery — After patching, validate that wolfSSL version 5.9.1 or later is confirmed active on all previously affected systems. Re-test TLS handshake behavior on critical services. Revoke and reissue any certificates that may have been in scope during a potential exploitation window if compromise is suspected. Resume normal network exposure for previously isolated devices only after patch confirmation.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SC-17 (Public Key Infrastructure Certificates), NIST SC-23 (Session Authenticity), NIST SI-6 (Security and Privacy Function Verification), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Validate patched TLS behavior using `testssl.sh` (free, open-source) against each restored service endpoint: `./testssl.sh --severity HIGH --certnames :` will confirm certificate chain validation is functioning correctly and flag any residual handshake anomalies. For certificate revocation on a constrained PKI budget, if your internal CA uses OCSP, use `openssl ocsp -issuer chain.pem -cert suspect_cert.pem -url ` to confirm revocation status of potentially-in-scope certificates before trusting them again. For IoT and ICS devices restored to network, capture a 24-hour post-recovery PCAP baseline using tcpdump and compare TLS session establishment rates and certificate fingerprints against the pre-incident baseline to detect any persistent MITM sessions that survived the patch window.

Evidence: Before resuming normal network exposure, collect a post-patch TLS certificate fingerprint snapshot using `openssl s_client -connect : 2>/dev/null | openssl x509 -fingerprint -noout -sha256` for each critical service and compare against the pre-patch baseline captured in Step 1. Any certificate fingerprint change on a device that was not intentionally reissued is a high-fidelity indicator that a forged certificate was accepted and may still be cached or in active use by a downstream client. Retain this comparison log as a recovery validation artifact per NIST AU-11 (Audit Record Retention) requirements.

Step 5: Post-Incident — Audit SBOM and firmware dependency tracking processes to determine how wolfSSL version exposure was discovered and how quickly. Evaluate gaps in third-party library visibility across embedded and IoT device inventories. Establish a recurring firmware dependency review cadence and confirm that downstream vendor patch notifications are covered by your vulnerability management program. Document devices where patching will remain delayed and define compensating controls and review dates for each.

NIST Phase: Post Incident

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

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST SI-2 (Flaw Remediation), NIST SI-5 (Security Alerts, Advisories, and Directives), 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 2.1 (Establish and Maintain a Software Inventory)

Compensating: Integrate Syft or CycloneDX SBOM generation into your firmware build pipeline using a pre-commit hook or CI step so that every firmware image automatically produces a machine-readable SBOM; store these in a version-controlled repository so that future wolfSSL-class vulnerabilities can be scoped in minutes rather than days. For ongoing vendor firmware tracking without a commercial solution, create a structured watchlist in a free VulnDB feed or NVD API query (`curl 'https://services.nvd.nist.gov/rest/json/cves/2.0?keywordSearch=wolfSSL'`) on a weekly cron job that alerts when new CVEs reference wolfSSL, CyaSSL, or your specific device vendor names. For each device with a documented patch delay, create a plain-text compensating control record with: device identifier, wolfSSL

version confirmed, isolation status, compensating control description, review date (not to exceed 30 days), and approving authority.

Evidence: Compile a final incident timeline artifact documenting: (1) the date wolfSSL 5.9.1 was released (April 8, 2026), (2) the date each affected device was identified, (3) the date each device was patched or isolated, and (4) the mean time to remediate (MTTR) per device class (IoT vs. ICS vs. router). This timeline serves as evidence for regulatory reporting if any affected device processed PII or PHI, and as a lessons-learned baseline to measure SBOM visibility improvement against the next embedded library CVE. Retain all PCAP captures, certificate baseline snapshots, firmware hash manifests, and compensating control registers per your documented retention policy under NIST AU-11 (Audit Record Retention).

Detection Guidance

Primary detection method is inventory-based: search asset management systems, software composition analysis (SCA) tools, and firmware SBOM records for wolfSSL or CyaSSL version strings below 5.9.1. On Linux-based embedded systems, run 'dpkg -l | grep wolfssl' or 'rpm -qa | grep wolfssl' where package management is available. For compiled firmware, use binary analysis tools to identify wolfSSL version markers in firmware images. Network-side detection is limited without specific IDS signatures; monitor for TLS handshakes from devices presenting certificates with anomalous or truncated digest fields if your TLS inspection capability supports deep certificate field analysis. No confirmed public exploit code or IOC patterns were available as of April 2026; check NVD at https://nvd.nist.gov/vuln/detail/CVE-2026-5194 (authoritative source) and wolfSSL's official GitHub releases page (verify the domain is github.com/wolfSSL/) for updated detection signatures.

Framework Mappings

MITRE-ATTACK

- **T1553.004** — Install Root Certificate
- **T1600** — Weaken Encryption
- **T1600.001** — Reduce Key Space
- **T1557.002** — ARP Cache Poisoning
- **T1557** — Adversary-in-the-Middle

OWASP-TOP10-2021

- **A02:2021** — Cryptographic Failures
- **A07:2021** — Identification and Authentication Failures

NIST-800-53R5

- **SC-8** — Transmission Confidentiality and Integrity
- **SC-17** — Public Key Infrastructure Certificates
- **IA-2** — Identification and Authentication (Organizational Users)
- **SC-13** — Cryptographic Protection

CIS-V8

- **3.10** — Encrypt Sensitive Data in Transit

- **6.3** — Require MFA for Externally-Exposed Applications
- **7.3** — Perform Automated Operating System Patch Management
- **7.4** — Perform Automated Application Patch Management

HIPAA-SECURITY

- **164.312(d)** — Person or Entity Authentication
- **164.312(e)(1)** — Transmission Security

SOC2-TSC

- **CC6.1** — Logical access security software, infrastructure, and architectures
- **CC9.2** — Manages risks associated with vendors and business partners

ISO-27001-2022

- **A.8.8** — Management of technical vulnerabilities
- **A.5.34** — Privacy and protection of personal information
- **A.5.21** — Managing information security in the ICT supply chain
- **A.8.24** — Use of cryptography

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1553.004	Install Root Certificate	Defense-Evasion
T1600	Weaken Encryption	Defense-Evasion
T1600.001	Reduce Key Space	Defense-Evasion
T1557.002	ARP Cache Poisoning	Credential-Access
T1557	Adversary-in-the-Middle	Credential-Access

Sources

Source	URL	Tier
Security News	https://www.bleepingcomputer.com/news/security/critical-flaw-in-wol...	T3
CVE-2026-5194 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-5194	T1
CVE-2026-5194 - Red Hat Customer Portal	https://access.redhat.com/security/cve/cve-2026-5194	T3
CVE-2026-5194 | Mondoo Vulnerability Intelligence	https://mondoo.com/vulnerability-intelligence/vulnerability/CVE-202...	T3

Source	URL	Tier
CVE-2026-5194 Security Vulnerability Analysis & Exploit Details	https://cve.akaoma.com/cve-2026-5194	T3

Generated 2026-04-13 18:26 UTC by TJS Security Command Center