INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-14 06:52 UTC

CVE-2026-42945: 18-Year-Old NGINX Rewrite Module Heap Overflow Enables Unauthenticated RCE

CVE VULNERABILITY | **CRITICAL** | CVSS 9.5

SCC Item ID	SCC-CVE-2026-0174
Type	CVE Vulnerability
CVE ID	CVE-2026-42945, CVE-2026-42946, CVE-2026-40701, CVE-2026-42934
Severity	CRITICAL
CVSS Base Score	9.5
Affected Products	NGINX Plus R32-R36; NGINX Open Source 1.0.0-1.30.0; NGINX Open Source 0.6.27-0.9.7 (no fix planned); NGINX Instance Manager 2.16.0-2.21.1; F5 WAF for NGINX 5.9.0-5.12.1; NGINX App Protect WAF 4.9.0-4.16.0 and 5.1.0-5.8.0; F5 DoS for NGINX 4.8.0; NGINX App Protect DoS 4.3.0-4.7.0; NGINX Gateway Fabric 1.3.0-1.6.2 and 2.0.0-2.5.1; NGINX Ingress Controller 3.5.0-3.7.2, 4.0.0-4.0.1, and 5.0.0-5.4.1
Published	2026-05-14T02:00:09
Discovery Source	Rss

Executive Summary

A critical heap overflow vulnerability, present in NGINX's rewrite module for approximately 18 years, allows an unauthenticated attacker to execute arbitrary code on affected servers with a single HTTP request. NGINX serves approximately 34% of global web infrastructure, placing critical infrastructure at risk. The flaw affects NGINX Open Source, NGINX Plus, and a broad F5 product ecosystem including WAF, Ingress Controller, and Gateway Fabric components. A public proof-of-concept exploit is already available, meaning exploitation is accessible to low-skilled attackers, and active scanning or attacks should be assumed imminent.

Technical Analysis

CVE-2026-42945 is a heap buffer overflow (CWE-122) in NGINX's ngx_http_rewrite_module. A single unauthenticated HTTP request can trigger heap corruption, enabling RCE on systems where ASLR is disabled or can be bypassed. CVSS base score: 9.5 (Critical). Related CVEs, CVE-2026-42946, CVE-2026-40701, and CVE-2026-42934, indicate additional weaknesses in the same or adjacent modules. CWE cluster includes CWE-125 (out-of-bounds read), CWE-416 (use-after-free), and CWE-789 (uncontrolled memory allocation).

MITRE techniques: T1190 (Exploit Public-Facing Application), T1068 (Exploitation for Privilege Escalation), T1059 (Command and Scripting Interpreter), T1203 (Exploitation for Client Execution), T1499 (Endpoint Denial of Service). Affected versions: NGINX Plus R32-R36; NGINX Open Source 1.0.0-1.30.0; NGINX Open Source 0.6.27-0.9.7 (end-of-life, no fix available); NGINX Instance Manager 2.16.0-2.21.1; F5 WAF for NGINX 5.9.0-5.12.1; NGINX App Protect WAF 4.9.0-4.16.0 and 5.1.0-5.8.0; F5 DoS for NGINX 4.8.0; NGINX App Protect DoS 4.3.0-4.7.0; NGINX Gateway Fabric 1.3.0-1.6.2 and 2.0.0-2.5.1; NGINX Ingress Controller 3.5.0-3.7.2, 4.0.0-4.0.1, and 5.0.0-5.4.1. A public PoC (Nginx-Rift) has been published by DepthFirstDisclosures on GitHub. Vendor advisory: F5 K000161019. Legacy versions 0.6.27-0.9.7 will not receive patches; isolation or replacement is required.

Action Checklist

1. Immediate Containment: Identify all internet-facing NGINX and F5 instances running affected versions (NGINX Plus R32-R36, NGINX Open Source 1.0.0-1.30.0, affected F5 product ranges per K000161019). Deploy WAF or IPS rules blocking HTTP requests with excessively long URI segments or rewrite-module-specific payloads matching Nginx-Rift PoC signatures. Restrict external access to unpatched instances where operationally feasible. Isolate NGINX Open Source 0.6.27-0.9.7 instances immediately; no patch will be issued.
2. Detection: Query web server access logs and SIEM for anomalous HTTP requests with oversized or malformed URI rewrite patterns against NGINX endpoints. Look for unexpected process spawning from nginx worker processes (e.g., bash, sh, curl, wget as child PIDs of nginx). Review EDR telemetry for heap spray indicators or unusual memory allocation patterns on NGINX hosts. Monitor for inbound connections from IPs associated with the Nginx-Rift PoC repository and known scanning infrastructure.
3. Eradication: Apply vendor patches per F5 advisory K000161019: upgrade NGINX Plus to R37 or later; upgrade NGINX Open Source to 1.31.0 or later; update F5 WAF, App Protect WAF, App Protect DoS, Gateway Fabric, and Ingress Controller to the fixed versions specified in K000161019. For instances running end-of-life versions 0.6.27-0.9.7, decommission or replace; no patch path exists.
4. Recovery: After patching, verify running NGINX version via 'nginx -v' on all hosts and confirm no older binaries remain in container images or deployment pipelines. Re-enable production traffic incrementally, monitoring access logs and application error rates for anomalies. Confirm ASLR is enabled on all NGINX hosts (sysctl kernel.randomize_va_space should return 2 on Linux). Review container base images in CI/CD for bundled NGINX versions and rebuild from patched base.
5. Post-Incident: Audit NGINX version governance: establish a policy requiring all instances to be tracked in asset inventory with version and patch status. Evaluate whether NGINX instances running without ASLR represent a systemic configuration gap and enforce ASLR as a baseline hardening requirement. Review ingress and edge exposure for any server software running end-of-life versions with no patch path; this incident pattern (18-year-old unpatched code) warrants a broader legacy software audit.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE

Escalation Criteria	Escalate to CISO and legal counsel immediately if nginx error.log, auditd records, or EDR telemetry on any internet-facing host shows evidence of successful heap overflow exploitation (segfault in nginx worker, unauthorized child process spawning, or post-exploitation network callback) — particularly for hosts processing PII, PHI, or cardholder data, which may trigger breach notification obligations under GDPR Article 33, HIPAA §164.412, or PCI DSS Requirement 12.10.5; additionally escalate if any NGINX Open Source 0.6.27–0.9.7 instance cannot be isolated within 4 hours, as no remediation path exists and continued internet exposure represents unacceptable residual risk given the public Nginx-Rift PoC availability.
Recovery Notes	After patching all instances to NGINX Plus R37 or NGINX Open Source 1.31.0 and confirming fixed F5 component versions per K000161019, monitor nginx access.log and error.log continuously for a minimum of 72 hours post-recovery, specifically watching for HTTP 500 responses on rewrite-heavy endpoints, unexpected nginx worker process restarts (indicating continued crash-loop exploitation attempts), or outbound connections from nginx worker PIDs to non-whitelisted destinations. Validate that all container images in active deployment pipelines have been rebuilt from patched base layers by re-running Grype or Trivy against the registry and confirming zero findings for CVE-2026-42945 before declaring recovery complete. Any NGINX Open Source 0.6.27–0.9.7 instance that could not be decommissioned must remain isolated from internet-facing exposure indefinitely with compensating network controls documented as a formal risk acceptance.
Forensic Artifacts	nginx access.log entries with oversized or structurally malformed URI segments targeting rewrite-processed paths — the heap overflow in ngx_http_rewrite_module is triggered by a crafted HTTP request, so the exploit payload will appear as an anomalous GET or POST URI in the access log, typically with response code 500 (crash) or unexpectedly 200 if exploitation succeeded before the worker crashed | nginx error.log entries referencing worker process segmentation faults, signal 11 (SIGSEGV), or heap corruption — a heap overflow exploit against the rewrite module will frequently cause worker process crashes logged here as 'worker process exited on signal 11' before a successful exploitation stabilizes | auditd EXECVE records showing processes with ParentPID matching nginx worker PIDs executing shells (bash, sh, dash), download utilities (curl, wget), or interpreters (python, perl) — this is the primary forensic indicator that CVE-2026-42945 RCE was successfully achieved and a post-exploitation command was executed | /proc//maps memory map captures taken during or immediately after suspected exploitation, showing anomalous executable heap regions or injected shared library mappings that would indicate heap spray shellcode staging specific to this vulnerability class | Network flow or pcap data capturing outbound connections initiated from nginx worker process PIDs to external IPs on non-standard ports, particularly short-lived connections consistent with reverse shell callbacks or C2 beaconing — the unauthenticated single-request RCE nature of CVE-2026-42945 means a successful exploit would immediately initiate attacker-controlled network activity from the nginx process user context

Per-Action IR Details

Containment — Identify all internet-facing NGINX and F5 instances running affected versions (NGINX Plus R32–R36, NGINX Open Source 1.0.0–1.30.0, affected F5 product ranges per F5 advisory K000161019). Place edge WAF or IPS rules blocking malformed rewrite-targeting HTTP requests. Restrict external access to unpatched instances where operationally feasible. Isolate NGINX Open Source 0.6.27–0.9.7 instances immediately — no patch will be issued.

NIST Phase: Containment

Reference: NIST 800-61r3 §3.3 — Containment, Eradication, and Recovery: Containment Strategy

Controls: NIST IR-4 (Incident Handling), NIST SC-7 (Boundary Protection), NIST CM-7 (Least Functionality), 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: Use 'nginx -v' or 'nginx -V' on each host via an SSH loop script to enumerate all deployed versions: for h in $(cat hosts.txt); do ssh $h 'nginx -v 2>&1'; done. For container environments, run: docker ps --format '{{.Image}}' | xargs -I{} docker run --rm {} nginx -v 2>&1. Block malformed rewrite-targeting requests at the network perimeter using iptables with string-match rules: iptables -I INPUT -p tcp --dport 80 -m string --string 'PATTERN' --algo bm -j DROP. Deploy a Sigma rule monitoring for curl/wget/bash spawned as child processes of nginx worker PIDs using auditd: auditctl -a always,exit -F arch=b64 -S execve -F ppid=$(pgrep -d, nginx) -k nginx_child_exec.

Evidence: Before isolating any instance, capture: (1) full 'nginx -V' output including compiled module list confirming rewrite module presence; (2) running process tree snapshot via 'ps auxf' or 'pstree -p' to capture any live child processes already spawned by nginx workers; (3) active network connections from nginx worker PIDs via 'ss -tulnp | grep nginx' and 'lsof -p $(pgrep nginx) -i'; (4) current nginx.conf and all included rewrite rule configurations from /etc/nginx/ recursively; (5) timestamp-preserved copy of /var/log/nginx/access.log and error.log prior to any log rotation or truncation.

Detection — Query web server access logs and SIEM for anomalous HTTP requests with oversized or malformed URI rewrite patterns against NGINX endpoints. Look for unexpected process spawning from nginx worker processes (e.g., bash, sh, curl, wget as child PIDs of nginx). Review EDR telemetry for heap spray indicators or unusual memory allocation patterns on NGINX hosts. Monitor for inbound connections from IPs associated with the Nginx-Rift PoC repository and known scanning infrastructure.

NIST Phase: Detection Analysis

Reference: NIST 800-61r3 §3.2 — Detection and Analysis: Signs of an Incident and Sources of Precursors and Indicators

Controls: NIST IR-5 (Incident Monitoring), NIST SI-4 (System Monitoring), NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-12 (Audit Record Generation), CIS 8.2 (Collect Audit Logs)

Compensating: Parse nginx access logs for oversized rewrite-pattern requests using: awk 'length($7) > 512 || $7 ~ /(\.\.|%2e%2e|%00|\\x[0-9a-f]{2})/' /var/log/nginx/access.log. Enable auditd to catch nginx worker process spawning unauthorized children: auditctl -a always,exit -F arch=b64 -S execve -F ppid=$(cat /var/run/nginx.pid) -k nginx_rce_spawn; ausearch -k nginx_rce_spawn. Deploy Sysmon on Windows-hosted NGINX (if applicable) with Event ID 1 (Process Create) filtered for ParentImage containing 'nginx.exe' and Image containing 'cmd.exe', 'powershell.exe', or 'curl.exe'. Use goatcounter or manual grep to identify Nginx-Rift PoC scanner IPs: grep -E '(GET|POST|HEAD)' /var/log/nginx/access.log | awk '{print $1}' | sort | uniq -c | sort -rn | head -50, then cross-reference top IPs against the Shodan InternetDB API or AbuseIPDB free tier.

Evidence: Capture before any log rotation: (1) nginx access.log entries showing the specific HTTP method, URI path length, rewrite-module-targeted path segments, response code (look for 500 or unexpected 200 on attack paths), and source IP; (2) nginx error.log entries referencing heap allocation failures, segmentation faults, or rewrite rule processing errors that indicate the overflow was triggered; (3) auditd or /proc//fd snapshot showing file descriptors opened by nginx worker processes immediately following suspicious requests; (4) kernel dmesg output for segfault or heap corruption messages referencing nginx worker PIDs; (5) /proc//maps memory map to identify anomalous mapped regions indicative of heap spray or injected shellcode staging.

Eradication — Apply vendor patches per F5 advisory K000161019: upgrade NGINX Plus to R37 or later; upgrade NGINX Open Source to 1.31.0 or later; update F5 WAF, App Protect WAF, App Protect DoS, Gateway Fabric, and Ingress Controller to the fixed versions specified in K000161019. For instances running EOL versions 0.6.27–0.9.7, decommission or replace — no patch path exists.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Containment, Eradication, and Recovery: Eradication and Recovery

Controls: NIST SI-2 (Flaw Remediation), NIST SI-3 (Malicious Code Protection), NIST CM-7 (Least Functionality), 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 teams without an enterprise patch management platform, script the upgrade sequence using the official nginx.org package repository: on Debian/Ubuntu, run 'apt-get install --only-upgrade nginx=1.31.0*' pinned to the

fixed version; on RHEL/CentOS, run 'yum update nginx-1.31.0'. For NGINX Plus R37, follow F5's yum/apt repository swap procedure documented in K000161019 — do not pull from unofficial mirrors. For EOL 0.6.27–0.9.7 instances with no patch path, verify decommission by checking for lingering nginx processes post-shutdown: 'pgrep -a nginx' and 'ss -tlnp | grep :80' to confirm no listener remains. Validate rebuilt container images contain no embedded vulnerable nginx binary by running: docker run --rm nginx -v and confirming output shows 1.31.0 or NGINX Plus R37.

Evidence: Before patching, preserve full forensic snapshots for any host showing signs of exploitation: (1) complete filesystem snapshot or disk image of /usr/sbin/nginx, /etc/nginx/, and /var/log/nginx/ with hash verification (sha256sum); (2) memory dump of all running nginx worker processes using 'gcore ' or '/proc//mem' capture for post-incident heap analysis; (3) copy of all active rewrite rule configurations from nginx.conf and conf.d/ that may have triggered the overflow — these define the specific rewrite directives that made the host exploitable; (4) list of all loaded nginx modules ('nginx -V 2>&1 | grep modules') to confirm ngx_http_rewrite_module was compiled in and active.

Recovery — After patching, verify running NGINX version via 'nginx -v' on all hosts and confirm no older binaries remain in container images or deployment pipelines. Re-enable production traffic incrementally, monitoring access logs and application error rates for anomalies. Confirm ASLR is enabled on all NGINX hosts (sysctl kernel.randomize_va_space should return 2 on Linux). Review container base images in CI/CD for bundled NGINX versions and rebuild from patched base.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Containment, Eradication, and Recovery: Recovery

Controls: NIST SI-7 (Software, Firmware, and Information Integrity), NIST SI-6 (Security and Privacy Function Verification), NIST CM-7 (Least Functionality), NIST CP-10 (System Recovery and Reconstitution), CIS 4.6 (Securely Manage Enterprise Assets and Software), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Automate post-patch version verification across all hosts with: for h in $(cat nginx_hosts.txt); do echo "$h: $(ssh $h 'nginx -v 2>&1')"; done > nginx_version_audit_$(date +%Y%m%d).txt. Scan CI/CD pipeline container registries for embedded vulnerable nginx using Grype (free, by Anchore): 'grype : --only-fixed' — any finding referencing CVE-2026-42945 in nginx indicates the base image has not been rebuilt from the patched layer. Verify ASLR on each Linux host: ssh $h 'sysctl kernel.randomize_va_space' — value must be 2; if 0 or 1, enforce immediately via 'sysctl -w kernel.randomize_va_space=2' and persist in /etc/sysctl.d/99-aslr.conf. Monitor nginx error.log for a 72-hour post-recovery window for any recurrence of heap allocation errors that may indicate re-exploitation attempts: tail -f /var/log/nginx/error.log | grep -iE '(heap|segfault|signal 11|rewrite)'.

Evidence: Capture post-recovery validation artifacts: (1) output of 'nginx -V' on every restored host, timestamped and hashed, confirming version 1.31.0 or NGINX Plus R37; (2) 'sysctl kernel.randomize_va_space' output from each host confirming value of 2, as ASLR at level 2 was a key mitigation factor limiting exploit reliability against this heap overflow — document any hosts where ASLR was previously disabled (value 0 or 1) as they represent highest prior exploitation risk; (3) container image manifest digests from CI/CD registry after rebuild, confirming no layer references the vulnerable nginx base; (4) nginx access.log and error.log samples from the first 24 hours of restored production traffic as baseline for anomaly comparison.

Post-Incident — Audit NGINX version governance: establish a policy requiring all instances to be tracked in asset inventory with version and patch status. Evaluate whether NGINX instances running without ASLR represent a systemic configuration gap and enforce ASLR as a baseline hardening requirement. Review ingress and edge exposure for any server software running EOL versions with no patch path — this incident pattern (18-year-old unpatched code) warrants a broader legacy software audit.

NIST Phase: Post Incident

Reference: NIST 800-61r3 §4 — Post-Incident Activity: Lessons Learned and Using Collected Incident Data

Controls: NIST IR-4 (Incident Handling), NIST IR-8 (Incident Response Plan), NIST RA-3 (Risk Assessment), NIST CM-8 (System Component Inventory), NIST SI-2 (Flaw Remediation), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 7.2 (Establish and Maintain a Remediation Process)

Compensating: Establish ongoing NGINX version governance using osquery: deploy the query 'SELECT name, version, source FROM deb_packages WHERE name LIKE "%nginx%" UNION SELECT name, version, source FROM

rpm_packages WHERE name LIKE "%nginx%";' as a scheduled osquery pack running daily, with results exported to a CSV and diffed against the authorized version list. For the broader legacy software audit triggered by this 18-year-old vulnerability pattern, run Grype or Trivy against all container images in the registry to surface any software with no vendor support path: 'trivy image --ignore-unfixed=false | grep -i "end-of-life"'. Document NGINX Open Source 0.6.27–0.9.7 instances decommissioned during this incident in the asset inventory with disposition date and rationale, per NIST CM-8 requirements. Enforce ASLR as a CIS Benchmark Level 1 Linux baseline control by adding kernel.randomize_va_space=2 to all Ansible/Chef/Puppet hardening playbooks.

Evidence: Preserve for lessons learned and potential regulatory reporting: (1) complete timeline of CVE-2026-42945 exposure window per asset — from nginx install date or last verified version check through patch application date — this establishes organizational risk exposure duration for any breach notification assessment; (2) list of NGINX Open Source 0.6.27–0.9.7 instances that were decommissioned, including their network exposure (internet-facing vs. internal), as these represent the highest-risk population given the absence of any patch path; (3) ASLR configuration state (sysctl kernel.randomize_va_space) for all NGINX hosts as of incident discovery — hosts with ASLR disabled (value 0) during the exposure window had materially higher exploit reliability and should be flagged in the incident record; (4) CI/CD pipeline scan results showing which container base images contained the vulnerable nginx version and how long those images were in production rotation.

Detection Guidance

Check NGINX access logs for requests with unusually long or structurally malformed URI segments, particularly those targeting endpoints processed by rewrite rules. On Linux hosts, monitor for nginx worker processes spawning unexpected child processes (shell interpreters, network tools) using process tree visibility in EDR or auditd rules on execve syscalls with ppid matching nginx workers. In SIEM, correlate HTTP 500-series errors spiking on NGINX hosts with concurrent process anomalies; heap corruption often produces crash-restart cycles before successful exploitation. Review network flow data for unexpected outbound connections from NGINX hosts, which may indicate post-exploitation callback activity. Container environments should audit running image digests against patched base image hashes to confirm no unpatched NGINX binaries remain deployed. Deploy IDS/WAF signatures from vendor feeds referencing CVE-2026-42945.

Indicators of Compromise

Type	Value	Context	Confidence
URL	`https://github.com/DepthFirstDisclosures/Nginx-Rift`	Public proof-of-concept exploit for CVE-2026-42945 published by DepthFirstDisclosures. Treat inbound requests from IPs associated with this repository's users or known PoC scanners as high-suspicion.	MEDIUM

Framework Mappings

MITRE-ATTACK

- **T1068** — Exploitation for Privilege Escalation
- **T1499** — Endpoint Denial of Service
- **T1059** — Command and Scripting Interpreter

- **T1190** — Exploit Public-Facing Application
- **T1203** — Exploitation for Client Execution

NIST-800-53R5

- **AC-6** — Least Privilege
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **SC-5** — Denial-of-Service Protection
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **SI-7** — Software, Firmware, and Information Integrity
- **CA-8** — Penetration Testing
- **RA-5** — Vulnerability Monitoring and Scanning
- **SI-16** — Memory Protection

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1068	Exploitation for Privilege Escalation	Privilege-Escalation
T1499	Endpoint Denial of Service	Impact
T1059	Command and Scripting Interpreter	Execution
T1190	Exploit Public-Facing Application	Initial-Access
T1203	Exploitation for Client Execution	Execution

Sources

Source	URL	Tier
Security News	https://thehackernews.com/2026/05/18-year-old-nginx-rewrite-module-...	T3

Source	URL	Tier
NGINX ngx_http_rewrite_module vulnerability CVE-2026-42945	https://my.f5.com/manage/s/article/K000161019	T3
CVE-2026-42946 - CVE Record	https://www.cve.org/CVERecord?id=CVE-2026-42946	T3
NGINX ngx_http_rewrite_module vulnerability CVE-2026-42945	https://seclists.org/oss-sec/2026/q2/519	T3
DepthFirstDisclosures/Nginx-Rift: exploit for CVE-2026-42945 - GitHub	https://github.com/DepthFirstDisclosures/Nginx-Rift	T3
NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-42945, CVE-2026-42946, CV...	T1

Generated 2026-05-14 06:52 UTC by TJS Security Command Center