INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-16 19:14 UTC

Widget Factory Joomla Content Editor - Widget Factory Joomla Content Editor Improper Access Control Vulnerability

CVE VULNERABILITY | CRITICAL | CVSS 9.8 | CISA KEV

SCC Item ID	SCC-CVE-2026-0310
Type	CVE Vulnerability
CVE ID	CVE-2026-48907
Severity	CRITICAL
CVSS Base Score	9.8
EPSS Score	0.0084 (53th percentile)
KEV Status	Yes — CISA Known Exploited Vulnerability (due: 2026-06-19)
Affected Products	Widget Factory Joomla Content Editor (version unspecified in available data)
Published	2026-06-16
Discovery Source	Cisa Kev

Executive Summary

A critical unauthenticated remote code execution vulnerability in the Widget Factory Joomla Content Editor plugin allows attackers to create rogue editor profiles and execute arbitrary PHP code on affected web servers. CISA has confirmed active exploitation in the wild and assigned a remediation due date of June 19, 2026. Any organization running this plugin on an internet-facing Joomla instance faces full server compromise without requiring any user interaction or credentials.

Technical Analysis

CVE-2026-48907 is a CWE-284 (Improper Access Control) vulnerability in the Widget Factory Joomla Content Editor plugin. An unauthenticated remote attacker can invoke the profile-creation endpoint without authentication, upload a PHP webshell or malicious file through the newly created editor profile, and achieve remote code execution on the underlying web server. This maps directly to MITRE ATT&CK T1190 (Exploit Public-Facing Application) for initial access and T1505.003 (Server Software Component: Web Shell) for persistence. CVSS base score is 9.8 (Critical). EPSS score is 0.00836 (52.8th percentile). The vulnerability appears in the CISA Known Exploited Vulnerabilities catalog with a federal remediation deadline of 2026-06-19. The specific affected version range was not available in the source data. NVD record:

https://nvd.nist.gov/vuln/detail/CVE-2026-48907. CISA KEV: https://www.cisa.gov/known-exploited-vulnerabilities-catalog.

Action Checklist

1. Step 1: Containment. Immediately take any internet-facing Joomla instance running Widget Factory Joomla Content Editor offline or block external access to it via WAF or firewall rule. Disable the plugin at the Joomla administrator level if a safe version or patch is not yet available. Reference NIST AC-4 (Information Flow Enforcement) and CIS 4.4 (Implement and Manage a Firewall on Servers).
2. Step 2: Detection. Search web server access logs for unexpected POST requests to editor profile-creation endpoints. Look for newly created files with .php extensions in upload or media directories. Scan for webshell indicators: base64-encoded strings in PHP files, eval(), system(), passthru(), or shell_exec() function calls in recently modified files. Use file integrity monitoring controls (per NIST SI-4 and AU-6) to flag unauthorized modifications to server-side files. Reference NIST SI-4 (System Monitoring) and CIS 8.2 (Collect Audit Logs).
3. Step 3: Eradication. Apply the vendor-supplied patch or upgrade to a remediated version of Widget Factory Joomla Content Editor as soon as one is available from the vendor. Until a patch exists, disable and remove the plugin entirely. Audit all files in Joomla upload and media directories for unauthorized PHP files and remove them. Rotate any credentials stored on the affected server or accessible from it, per NIST IA-5 (Authentication and Authorization).
4. Step 4: Recovery. After patching or removing the plugin, verify no webshells remain by running a full file integrity scan across the Joomla document root. Restore from a known-clean backup if unauthorized PHP files are found. Monitor web server logs and application logs continuously for 30 days post-remediation for signs of re-exploitation or lateral movement. Reference NIST AU-6 (Audit Record Review, Analysis, and Reporting).
5. Step 5: Post-Incident. Review third-party plugin inventory against CIS 2.1 (Establish and Maintain a Software Inventory) and CIS 2.2 (Ensure Authorized Software is Currently Supported). Assess whether the plugin-creation workflow exposed unnecessary file write permissions; apply NIST AC-6 (Least Privilege) and NIST IA-5 (Authentication and Authorization) to tighten web server file system permissions. Add unauthenticated profile-creation and PHP file upload patterns to detection rules as permanent hunts.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to CISO, legal counsel, and breach notification process immediately if forensic analysis confirms unauthorized PHP file execution occurred, database credential exposure is detected, or the affected Joomla instance stored, processed, or transmitted PII, PHI, or payment card data — given CVE-2026-48907's CVSS 9.8 rating, confirmed CISA KEV active exploitation status, and unauthenticated full-server-compromise blast radius.

Recovery Notes	Restoration must originate from a backup predating the earliest suspicious unauthenticated POST request identified in web server access logs; restoring a post-compromise backup risks reintroducing dropped webshells or rogue editor profiles. Post-restoration, run daily AIDE file integrity checks and grep-based log monitoring for PHP webshell function patterns (eval, base64_decode, shell_exec, passthru) for a minimum of 30 days. The Joomla instance should not be returned to internet-facing status until the Widget Factory Joomla Content Editor plugin is either fully patched to a vendor-confirmed remediated version or permanently removed and confirmed absent from the extension table.
Forensic Artifacts	Web server access logs (Apache: /var/log/apache2/access.log; Nginx: /var/log/nginx/access.log) — filter for unauthenticated POST requests (no Joomla session token or Authorization header) to Joomla component paths matching the Widget Factory Content Editor plugin registration or profile-creation endpoints, with HTTP 200 responses, as the primary exploitation IOC for CVE-2026-48907. | Joomla database table jos_extensions — rogue editor profiles created by the improper access control vulnerability will appear as unauthorized entries; compare against known-good extension manifest to identify attacker-inserted plugin registrations. | PHP files in Joomla upload and media directories (/var/www/html/images/, /var/www/html/media/, /var/www/html/tmp/) with creation or modification timestamps postdating the earliest suspicious access log entry — these represent webshells or remote access tools dropped via the arbitrary PHP code execution capability of CVE-2026-48907. | Web server process memory image (acquired via LiME or gcore targeting the Apache/PHP-FPM PID) — captures in-memory execution state of any PHP webshell payload, injected code, or active reverse shell connection established through the CVE-2026-48907 RCE before containment actions destroy live process state. | Web server error log (Apache: /var/log/apache2/error.log; Nginx: /var/log/nginx/error.log) and PHP error log — PHP execution errors, fatal errors, or warnings generated during attacker-controlled code execution will reference specific file paths and line numbers pointing directly to dropped webshell locations on the Joomla document root.

Per-Action IR Details

Step 1: Containment — Immediately take any internet-facing Joomla instance running Widget Factory Joomla Content Editor offline or block external access to it via WAF or firewall rule. Disable the plugin at the Joomla administrator level if a safe version or patch is not yet available. Reference NIST AC-4 (Information Flow Enforcement) and 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: If a WAF is unavailable, use iptables or ufw to immediately DROP all inbound traffic on ports 80/443 to the affected Joomla host: `sudo ufw deny in on eth0 to any port 80,443`. Alternatively, null-route the server's public IP at the router. Disable the plugin via direct Joomla database update if the admin panel is inaccessible: `UPDATE jos_extensions SET enabled=0 WHERE name='Widget Factory Joomla Content Editor';`

Evidence: Before blocking network access, capture volatile network state to preserve active attacker sessions or reverse shell connections this RCE may have established: run `netstat -ano` or `ss -tulnp` on the Joomla host to record all active TCP/UDP connections and listening services. Capture `who`, `w`, and `last` output to document any active or recent SSH/console sessions. Export current web server access logs (Apache: `/var/log/apache2/access.log`; Nginx: `/var/log/nginx/access.log`) and WAF logs in their live state before log rotation occurs. This exploit is unauthenticated, so active attacker sessions will show source IPs with no prior authentication events.

Step 2: Detection — Search web server access logs for unexpected POST requests to editor profile-creation endpoints. Look for newly created files with .php extensions in upload or media directories. Scan for webshell

indicators: base64-encoded strings in PHP files, eval(), system(), passthru(), or shell_exec() function calls in recently modified files. Use D3-SFA (System File Analysis) to flag unauthorized modifications to server-side files. Reference NIST SI-4 (System Monitoring) and CIS 8.2 (Collect Audit Logs).

NIST Phase: Detection Analysis

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

Controls: CIS 8.2 (Collect Audit Logs), NIST AU-6 (Audit Record Review, Analysis, And Reporting)

Compensating: Run the following grep against Apache/Nginx access logs to identify unauthenticated POST requests to the plugin's profile-creation endpoint: `grep -E 'POST.*(editor|profile|content-editor).*\.php' /var/log/apache2/access.log | grep -v '30[0-9]'`. To detect dropped webshells, run: `find /var/www/html -name '*.php' -newer /var/www/html/index.php -exec grep -lE '(eval|base64_decode|system|passthru|shell_exec)' {} \;`. Deploy a YARA rule scanning for PHP webshell function combinations in the Joomla media and upload directories: target strings `eval(base64_decode`, `system($_`, `passthru($_REQUEST`. Use ClamAV with the `--scan-mail=no --detect-pua` flags against the document root for rapid triage.

Evidence: Before scanning files (which updates atime metadata and may alter forensic timestamps), capture a complete recursive file listing with timestamps: `find /var/www/html -type f -printf '%T+ %p\n' | sort > /tmp/joomla_file_timeline_$(date +%F).txt`. Snapshot the Joomla database for rogue editor profiles: `mysqldump -u root -p joomla_db jos_extensions jos_users > /tmp/joomla_db_snapshot_$(date +%F).sql`. Preserve web server access logs and error logs (Apache error log: `/var/log/apache2/error.log`) before any log rotation. HTTP 200 responses to POST requests targeting `/administrator/components/com_contenteditor/` or similar plugin component paths with no prior authentication (no session cookie or Joomla token in the request headers) are the primary IOC for CVE-2026-48907 exploitation.

Step 3: Eradication — Apply the vendor-supplied patch or upgrade to a remediated version of Widget Factory Joomla Content Editor as soon as one is available from the vendor. Until a patch exists, disable and remove the plugin entirely. Audit all files in Joomla upload and media directories for unauthorized PHP files and remove them. Rotate any credentials stored on the affected server or accessible from it, per D3-CRO (Credential Rotation).

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: CIS 7.4 (Perform Automated Application Patch Management), CIS 7.2 (Establish and Maintain a Remediation Process), NIST AC-2 (Account Management)

Compensating: If no vendor patch exists yet, uninstall the Widget Factory Joomla Content Editor plugin via the Joomla Extension Manager and physically remove its directory: `rm -rf /var/www/html/plugins/editors/contenteditor /var/www/html/administrator/components/com_contenteditor`. Purge all PHP files in media and upload directories that are not part of the known-good Joomla core or other verified extensions: `find /var/www/html/images /var/www/html/media -name '*.php' -exec rm -v {} \;`. Rotate the Joomla database credentials in `configuration.php`, all Joomla Super User account passwords, and any server-level credentials (SSH keys, cPanel/hosting panel passwords, SFTP accounts) that were accessible from the compromised host.

Evidence: Before removing any files or rotating credentials, acquire a full memory image of the compromised web server process (Apache/PHP-FPM/Nginx worker) using LiME (Linux Memory Extractor) or `gcore ` targeting the web server PID to capture in-memory webshell execution context, injected code, or active reverse shell state. Record all currently open file handles by the web server process: `lsof -p $(pgrep apache2 | head -1) > /tmp/open_files_pre_eradication.txt`. Export the full Joomla `jos_extensions` and `jos_users` tables to document any rogue editor profiles created by CVE-2026-48907 exploitation before eradication destroys that evidence. Preserve SHA-256 hashes of all identified malicious PHP files before deletion: `sha256sum /path/to/webshell.php >> /tmp/ioc_hashes.txt`.

Step 4: Recovery — After patching or removing the plugin, verify no webshells remain by running a full file integrity scan across the Joomla document root. Restore from a known-clean backup if unauthorized PHP files are found. Monitor web server logs and application logs continuously for 30 days post-remediation for

signs of re-exploitation or lateral movement. Reference 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), CIS 4.4 (Implement and Manage a Firewall on Servers)

Compensating: Establish a file integrity baseline using `aide --init` on the recovered Joomla document root immediately post-restoration, then schedule `aide --check` daily via cron for the 30-day monitoring window to detect any re-dropped webshells. For log monitoring without a SIEM, deploy a cron-driven grep pipeline that alerts on PHP webshell function calls in new web server log entries: `grep -E '(eval|base64_decode|shell_exec|passthru)' /var/log/apache2/access.log | mail -s 'Joomla Webshell Alert' soc@example.com`. If restoring from backup, verify the backup predates the earliest suspicious POST request found in access logs during Step 2 analysis to avoid restoring a backdoored state.

Evidence: Before bringing the restored Joomla instance back online, generate and record SHA-256 hashes of all PHP files in the document root as the new integrity baseline: `find /var/www/html -name '*.php' -exec sha256sum {} \; > /tmp/joomla_clean_baseline_$(date +%F).txt`. Confirm the Joomla `jos_extensions` table contains no residual entries for Widget Factory Joomla Content Editor plugin components. Verify the web server process (Apache/Nginx) is not running as root and confirm file permissions on `/var/www/html` prevent web-process-writable PHP execution paths: `find /var/www/html -writable -name '*.php'` should return no results under a correctly hardened configuration.

Step 5: Post-Incident — Review third-party plugin inventory against CIS 2.1 (Establish and Maintain a Software Inventory) and CIS 2.2 (Ensure Authorized Software is Currently Supported). Assess whether the plugin-creation workflow exposed unnecessary file write permissions; apply NIST AC-6 (Least Privilege) and D3-UAP (User Account Permissions) to tighten web server file system permissions. Add unauthenticated profile-creation and PHP file upload patterns to detection rules as permanent hunts.

NIST Phase: Post Incident

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

Controls: CIS 2.1 (Establish and Maintain a Software Inventory), CIS 2.2 (Ensure Authorized Software is Currently Supported), NIST AC-6 (Least Privilege), NIST AU-2 (Event Logging)

Compensating: Enumerate all installed Joomla extensions (plugins, modules, components) and cross-reference against the Joomla Vulnerable Extensions List (VEL) and CISA KEV catalog: `SELECT name, type, enabled, manifest_cache FROM jos_extensions WHERE type IN ('plugin','component','module');`. Harden web server file system permissions so the PHP process user (e.g., `www-data`) cannot write to the document root outside designated upload paths: `chown -R root:www-data /var/www/html && find /var/www/html -type d -exec chmod 755 {} \; && find /var/www/html -type f -exec chmod 644 {} \;`. Write a permanent Sigma rule targeting POST requests with no Authorization header to Joomla component paths matching `com_contenteditor` or equivalent plugin registration endpoints, flagging HTTP 200 responses.

Evidence: Preserve the complete incident artifact package for lessons-learned documentation: web server access logs spanning the full exposure window, the file timeline snapshot from Step 2, SHA-256 hashes of all identified malicious PHP files, the rogue editor profile database dump, and memory acquisition from Step 3. This artifact set supports both internal post-incident review per NIST 800-61r3 §4 and any required regulatory breach notification analysis if the Joomla instance processed PII or PHI. Retain artifacts for a minimum period consistent with your organization's incident record retention policy and applicable regulatory requirements.

Detection Guidance

Query web server access logs for POST requests to any endpoint containing 'profile', 'editor', or 'upload' path segments from unauthenticated sessions (no valid session cookie or auth token). Alert on HTTP 200 responses to those endpoints from external IP addresses. Scan Joomla media, images, and tmp directories for .php files

created or modified within the last 72 hours; any PHP file in an upload directory is an indicator of compromise. Search for PHP function patterns associated with webshells: eval(base64_decode()), system(), passthru(), shell_exec(), and proc_open() in recently modified files. Review Joomla administrator logs for newly created editor profiles not initiated by a known administrator account. Apply file integrity monitoring controls (per NIST SI-4 and AU-6) to establish a baseline and alert on deviations. Cross-reference outbound connections from the web server process against expected destinations; unexpected outbound traffic from the PHP process may indicate a webshell receiving commands.

Framework Mappings

MITRE-ATTACK

- **T1505.003** — Web Shell
- **T1190** — Exploit Public-Facing Application

NIST-800-53R5

- **CM-2** — Baseline Configuration
- **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
- **SC-7** — Boundary Protection
- **SI-2** — Flaw Remediation
- **AC-3** — Access Enforcement
- **IR-5** — Incident Monitoring

OWASP-TOP10-2021

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

CIS-V8

- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control

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
T1505.003	Web Shell	Persistence
T1190	Exploit Public-Facing Application	Initial-Access

Sources

Source	URL	Tier
cisa_kev	https://www.cisa.gov/known-exploited-vulnerabilities-catalog	T1
CVE-2026-48907 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-48907	T1
CVE-2026-48907 - CVE Record	https://www.cve.org/CVERecord?id=CVE-2026-48907	T3
0xBlackash/CVE-2026-48907 - GitHub	https://github.com/0xBlackash/CVE-2026-48907	T3
CVE-2026-48907 - Vulnerability Details - OpenCVE	https://app.opencve.io/cve/CVE-2026-48907	T3

DISCLAIMER

This intelligence report is produced by Tech Jacks Solutions Security Command Center (SCC) for informational purposes only. It does not constitute professional security advice, legal counsel, or an incident response engagement. The information herein is derived from publicly available sources and AI-assisted analysis; while every effort is made to ensure accuracy, Tech Jacks Solutions makes no warranties regarding completeness or timeliness. Organizations should conduct their own validation and consult qualified security professionals before taking action based on this report. Tech Jacks Solutions is not liable for any damages resulting from the use of this information.

Generated 2026-06-16 19:14 UTC by TJS Security Command Center