INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-06-02 06:32 UTC

CVE-2026-7465: The Spectra Gutenberg Blocks - Website Builder for the Block Editor plugin for WordPress is vulnerab...

CVE VULNERABILITY | HIGH | CVSS 8.8

SCC Item ID	SCC-CVE-2026-0250
Type	CVE Vulnerability
CVE ID	CVE-2026-7465
Severity	HIGH
CVSS Base Score	8.8
EPSS Score	0.0008 (24th percentile)
Affected Products	Spectra Gutenberg Blocks - Website Builder for the Block Editor (WordPress plugin), all versions up to and including 2.19.25
Published	2026-05-30T10:16:23.860
Discovery Source	Nvd

Executive Summary

CVE-2026-7465 is a high-severity Remote Code Execution vulnerability in the Spectra Gutenberg Blocks WordPress plugin, affecting all versions through 2.19.25. Any authenticated user with Contributor-level access can execute arbitrary server-side PHP code without administrative privileges, giving attackers full control of the web server. Organizations running WordPress sites with this plugin installed face risks of complete site compromise, data theft, and infrastructure lateral movement.

Technical Analysis

CVE-2026-7465 (CVSS 8.8, CWE-269: Improper Privilege Management; CWE-94: Code Injection) affects Spectra Gutenberg Blocks plugin for WordPress, all versions up to and including 2.19.25. The attack chain requires only Contributor-level authentication. An attacker embeds a two-block payload in post content: the first block registers a fabricated block type using the legitimate 'uagb/' prefix with an attacker-controlled render_callback function pointer; the second block, using the same fake type, triggers PHP's call_user_func() to invoke that callback during sequential block rendering within the same page request. This yields arbitrary server-side PHP execution, equivalent to webshell-level access, without any admin credential requirement. MITRE ATT&CK mappings: T1190 (Exploit Public-Facing Application), T1546 (Event Triggered Execution), T1059.004 (Unix Shell). EPSS score is 0.0008 (23rd percentile), indicating low observed exploitation activity at

this time. CISA KEV: not listed. Patch status: update to a version above 2.19.25 when available from the plugin vendor.

Action Checklist

1. Step 1: Containment, Identify all WordPress instances in your environment running Spectra Gutenberg Blocks v2.19.25 or earlier (CIS 1.1: asset inventory). Immediately suspend Contributor-level post publishing permissions on affected sites until a patched version is confirmed available. Do not create new Contributor accounts during this window. If the site is internet-facing, consider placing it behind a WAF with rules blocking uagb/ prefix block registration patterns in POST request bodies.
2. Step 2: Detection, Query web server and WordPress application logs for POST requests to wp-admin/admin-ajax.php or wp-json/ endpoints originating from Contributor-level accounts containing 'uagb/' block registration patterns. Review audit logs (NIST AU-2, AU-6) for unusual PHP function invocations or unexpected outbound connections from the web server process. Behavioral indicator: PHP child processes spawned by the web server user without corresponding admin-initiated actions. No confirmed IOCs are available at this time.
3. Step 3: Eradication, Apply the vendor-released patch upgrading Spectra Gutenberg Blocks above version 2.19.25 (CIS 7.3, 7.4). Until a patch is available, enforce least privilege by removing Contributor-level roles from untrusted users (NIST AC-6, CIS 5.4). Disable the plugin if Contributor-level content submission from external or untrusted users is not a business requirement.
4. Step 4: Recovery, After patching, verify the installed plugin version via the WordPress admin dashboard or wp-cli ('wp plugin list'). Review all post content submitted by Contributor-level accounts since the site's last known-good state for embedded uagb/ block payloads. Monitor server process activity and outbound network connections for 30 days post-remediation (NIST SI-4). Rotate credentials for any accounts that had Contributor access or higher during the exposure window (D3-CRO).
5. Step 5: Post-Incident, Conduct a privilege audit across all WordPress installations to enforce minimum necessary roles (NIST AC-6, AC-2; D3-UAP). Implement a process for automated plugin patch management with monthly or more frequent cadence (CIS 7.4). Evaluate whether Contributor-level publishing requires editorial approval workflows to reduce the attack surface for future authenticated RCE classes.

IR / Forensic Enrichment

Triage Priority	URGENT
Escalation Criteria	Escalate immediately to CISO and legal counsel if forensic review of wp-admin/admin-ajax.php logs or the WordPress uploads directory reveals PHP webshells, confirmed reverse shell callbacks, or uagb/ block payloads successfully reaching 'published' status from Contributor accounts during the exposure window — any of these conditions indicate active exploitation and may trigger breach notification obligations if PII or PHI is stored on the affected WordPress instance.

Recovery Notes	After patching Spectra above v2.19.25, verify the fix by attempting to register a uagb/ block type via the wp-json/ REST API using a test Contributor account and confirming the server rejects the RCE-capable request. Conduct a full scan of all writable web directories (/wp-content/uploads/, /wp-content/themes/, /wp-content/plugins/) using ClamAV ('clamscan -r /var/www/html/wp-content/ --log=/var/log/clamav_post_patch.log') and a PHP webshell YARA ruleset for 30 days post-patch to detect any persistence mechanisms planted before remediation. Maintain heightened monitoring of outbound connections from the web server process (www-data/apache) for the same 30-day window, as a successful pre-patch RCE may have established a backdoor that survives plugin patching if a webshell was written to disk.
Forensic Artifacts	Web server access logs (Apache: /var/log/apache2/access.log; Nginx: /var/log/nginx/access.log) — filter for POST requests to wp-admin/admin-ajax.php and wp-json/wp/v2/blocks from Contributor-session source IPs during the exposure window; HTTP 200 responses to these endpoints are the primary exploitation indicator for CVE-2026-7465. | WordPress uploads directory (/wp-content/uploads/) and all plugin-writable directories — scan for PHP files created or modified after Spectra v2.19.25 was installed, as successful RCE via uagb/ block registration would most likely drop a PHP webshell to a writable web-accessible path for persistence. | WordPress database tables wp_posts and wp_postmeta — extract all content containing 'uagb/' block markup submitted by Contributor-role user IDs, including post revisions (post_type='revision'), to reconstruct what block payloads were registered and whether any contained PHP code execution attempts. | PHP-FPM or web server process tree and /tmp / /var/tmp directories — capture 'ps auxf' output and list all files in /tmp created by the www-data/apache/nginx user, as a PHP-based RCE exploit targeting Spectra would typically stage payloads or spawn reverse shells through the web server process user account. | WordPress debug log (/wp-content/debug.log, if WP_DEBUG_LOG=true) and PHP error log (/var/log/php*.log or as configured in php.ini) — these capture PHP warnings, fatal errors, and eval()/shell_exec() invocations that would appear if the Spectra block registration code path was abused to execute arbitrary PHP, providing a code-level execution trail absent from web server access logs.

Per-Action IR Details

Step 1: Containment — Identify all WordPress instances in your environment running Spectra Gutenberg Blocks v2.19.25 or earlier (CIS 1.1: asset inventory). Immediately restrict Contributor-level account creation and post publishing on affected sites until a patched version is confirmed available. If the site is internet-facing, consider placing it behind a WAF with rules blocking uagb/ prefix block registration patterns in POST request bodies.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST AC-6 (Least Privilege), NIST AC-2 (Account Management), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts)

Compensating: Run 'wp plugin list --name=ultimate-addons-for-gutenberg --fields=name,version,status --format=table' via WP-CLI across all WordPress installs (wrap in a bash loop if managing multiple sites). For WAF without budget: deploy ModSecurity with the OWASP Core Rule Set and add a custom rule matching REQUEST_URI containing 'admin-ajax.php' AND REQUEST_BODY containing 'uagb/' to block or alert. Alternatively, use Apache/Nginx access controls to restrict POST to wp-admin/admin-ajax.php by IP allowlist while the patch window is open.

Evidence: Before restricting accounts, snapshot the WordPress user table (wp_users and wp_usermeta) to preserve the current Contributor-role roster and their last-login timestamps. Export the wp_options table to capture active plugin

versions (option_name='active_plugins') as point-in-time evidence of the vulnerable Spectra plugin being active. Capture web server access logs (Apache: /var/log/apache2/access.log; Nginx: /var/log/nginx/access.log) covering the full exposure window from Spectra v2.19.25 installation date forward, focusing on POST requests to wp-admin/admin-ajax.php and wp-json/wp/v2/ endpoints.

Step 2: Detection — Query web server and WordPress application logs for POST requests to wp-admin/admin-ajax.php or wp-json/ endpoints originating from Contributor-level accounts containing 'uagb/' block registration patterns. Review audit logs (NIST AU-2, AU-6) for unusual PHP function invocations or unexpected outbound connections from the web server process. Behavioral indicator: PHP child processes spawned by the web server user without corresponding admin-initiated actions. No confirmed IOCs are available at this time.

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), NIST IR-5 (Incident Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Parse access logs with this command to surface suspicious Spectra block POST requests: grep -E 'POST.*(admin-ajax\.php|wp-json)' /var/log/nginx/access.log | grep -v '" [23]' | awk '{print $1, $7, $9}'. Install Auditd on the web server and add a rule to track process execution by the web server user: '-a always,exit -F arch=b64 -S execve -F uid=$(id -u www-data) -k php_exec'. Review /proc or use 'ps auxf' snapshots to identify unexpected child processes under the www-data/apache user tree. For PHP-level visibility, enable PHP's error_log and set 'log_errors = On' in php.ini to capture eval(), system(), shell_exec(), and passthru() invocations if error reporting is set to E_ALL.

Evidence: Collect WordPress debug log (/wp-content/debug.log if WP_DEBUG_LOG is enabled) for PHP errors or warnings generated during block registration abuse. Extract all POST entries to wp-admin/admin-ajax.php with HTTP 200 responses from Contributor-level session IPs — cross-reference source IPs against wp_usermeta for the 'contributor' capability assignment. Check /tmp, /var/tmp, and the WordPress uploads directory (/wp-content/uploads/) for newly created PHP files or shell scripts, which are the primary persistence artifacts from an authenticated RCE exploit. Review outbound network connections from the PHP-FPM or Apache process using 'ss -tupn' or 'netstat -tupn' and compare against a known-good baseline to detect reverse shell callbacks.

Step 3: Eradication — Apply the vendor-released patch upgrading Spectra Gutenberg Blocks above version 2.19.25 (CIS 7.3, 7.4). Until a patch is available, enforce least privilege by removing Contributor-level roles from untrusted users (NIST AC-6, CIS 5.4). Disable the plugin if Contributor-level content submission from external or untrusted users is not a business requirement.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST AC-6 (Least Privilege), NIST CM-7 (Least Functionality), CIS 7.3 (Perform Automated Operating System Patch Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts)

Compensating: Apply the patch via WP-CLI: 'wp plugin update ultimate-addons-for-gutenberg --version=[patched_version]' and verify with 'wp plugin get ultimate-addons-for-gutenberg --field=version'. If no patch is yet released, disable via WP-CLI: 'wp plugin deactivate ultimate-addons-for-gutenberg'. To bulk-demote Contributor accounts pending review: run this WP-CLI command 'wp user list --role=contributor --format=csv | tail -n +2 | cut -d, -f1 | xargs -I{} wp user set-role {} subscriber' — document each account before role change. Search /wp-content/uploads/ and all writable directories recursively for PHP files created or modified during the exposure window: 'find /var/www/html/wp-content/ -name "*.php" -newer /var/www/html/wp-config.php -ls'.

Evidence: Before removing any artifacts, forensically image or tar the WordPress uploads directory and any writable web directories: 'tar czf /evidence/uploads_$(date +%Y%m%d).tar.gz /var/www/html/wp-content/uploads/'. Preserve the complete WordPress database dump ('wp db export /evidence/db_pre_patch.sql') to retain wp_posts content submitted by Contributor accounts, which may contain serialized uagb/ block payloads used for RCE. Capture file system metadata (inode creation/modification timestamps) for any PHP files found in uploads/ or tmp/ before deleting them, as these timestamps establish the exploitation timeline.

Step 4: Recovery — After patching, verify the installed plugin version via the WordPress admin dashboard or wp-cli ('wp plugin list'). Review all post content submitted by Contributor-level accounts since the site's last known-good state for embedded uagb/ block payloads. Monitor server process activity and outbound network connections for 30 days post-remediation (NIST SI-4). Rotate credentials for all accounts that had Contributor access or higher during the exposure window (D3-CRO).

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST SI-2 (Flaw Remediation), NIST SI-4 (System Monitoring), NIST IA-5 (Authenticator Management), NIST CP-10 (System Recovery and Reconstitution), CIS 5.2 (Use Unique Passwords), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Query the WordPress database directly to surface all posts and revisions containing Spectra block markup from Contributor accounts during the exposure window: 'wp db query "SELECT ID, post_author, post_date, post_status, LEFT(post_content,200) FROM wp_posts WHERE post_content LIKE \'%uagb/%\' AND post_author IN (SELECT user_id FROM wp_usermeta WHERE meta_key=\'wp_capabilities\' AND meta_value LIKE \'%contributor%\') ORDER BY post_date DESC;"'. For 30-day post-remediation monitoring without SIEM, configure a daily cron job: '0 6 * * * find /var/www/html/wp-content/ -name "*.php" -newer /var/www/html/wp-login.php -ls >> /var/log/new_php_files.log' and review alerts daily. Force password reset for all Contributor-and-above accounts via WP-CLI: 'wp user list --role=contributor --format=ids | xargs -I{} wp user update {} --user_pass=$(openssl rand -base64 16)'.

Evidence: Verify plugin version post-patch and record as evidence: 'wp plugin get ultimate-addons-for-gutenberg --fields=name,version,status > /evidence/plugin_post_patch_$(date +%Y%m%d).txt'. Retain the pre-patch database export and compare post-patch wp_posts content to identify any uagb/ block payloads that were published and may have triggered RCE. Preserve all web server and PHP-FPM access/error logs from the 30-day monitoring window for potential regulatory or legal review; ensure log rotation does not purge these files (NIST AU-11).

Step 5: Post-Incident — Conduct a privilege audit across all WordPress installations to enforce minimum necessary roles (NIST AC-6, AC-2; D3-UAP). Implement a process for automated plugin patch management with monthly or more frequent cadence (CIS 7.4). Evaluate whether Contributor-level publishing requires editorial approval workflows to reduce the attack surface for future authenticated RCE classes.

NIST Phase: Post Incident

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

Controls: NIST AC-2 (Account Management), NIST AC-6 (Least Privilege), NIST CA-7 (Continuous Monitoring), NIST SI-2 (Flaw Remediation), CIS 7.4 (Perform Automated Application Patch Management), CIS 6.1 (Establish an Access Granting Process), CIS 6.2 (Establish an Access Revoking Process), CIS 7.1 (Establish and Maintain a Vulnerability Management Process)

Compensating: Automate WordPress plugin vulnerability monitoring by subscribing to the WPScan Vulnerability Database API (free tier available) and scheduling a weekly WPScan run: 'wpscan --url https://yoursite.com --api-token YOUR_TOKEN --format json > /var/log/wpscan_$(date +%Y%m%d).json'. Enforce editorial approval for all Contributor submissions by installing the free 'PublishPress' plugin or configuring the native WordPress editorial workflow to require Editor/Admin approval before any Contributor post reaches 'published' status — this eliminates the authenticated-but-unprivileged RCE attack surface for this entire vulnerability class. Export a monthly Contributor-role audit report via WP-CLI: 'wp user list --role=contributor --fields=ID,user_login,user_email,user_registered --format=csv > /reports/contributor_audit_$(date +%Y%m).csv' and review for dormant or unrecognized accounts (CIS 5.3).

Evidence: Document the full lessons-learned record: timeline from plugin installation of v2.19.25 through patch application, list of all Contributor accounts active during the exposure window, any confirmed or suspected exploitation indicators found during detection, and all remediation actions taken with timestamps. Retain the pre- and post-patch database exports, forensic file system captures, and web server logs as a complete evidentiary package for a minimum of 12 months (NIST AU-11) or per your organization's retention policy, whichever is longer. If PII was stored on the affected WordPress site, preserve this package to support breach notification obligations under applicable regulations.

Detection Guidance

Primary log sources: WordPress application logs, web server access logs (Apache/Nginx), and PHP error logs. Query for POST requests from Contributor-level authenticated sessions containing 'uagb/' strings in request bodies, particularly to block rendering or REST API endpoints. Look for PHP process anomalies: unexpected child processes, outbound connections to non-CDN/non-update IPs initiated by the web server user (www-data, apache, nginx). In SIEM, correlate WordPress user role assignments against post submission events followed by anomalous server-side execution. If PHP function logging is enabled, flag any call_user_func() or call_user_func_array() invocations originating from block rendering context. NIST AU-6 review of audit records for unauthorized access patterns is recommended. No confirmed IOC hashes, IPs, or domains are available for this CVE at this time; detection must rely on behavioral and structural indicators described above.

Framework Mappings

MITRE-ATTACK

- **T1190** — Exploit Public-Facing Application
- **T1546** — Event Triggered Execution
- **T1059.004** — Unix Shell

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
- **CM-7** — Least Functionality
- **SI-3** — Malicious Code Protection
- **SI-4** — System Monitoring
- **AC-6** — Least Privilege
- **SI-10** — Information Input Validation

OWASP-TOP10-2021

- **A01:2021** — Broken Access Control
- **A03:2021** — Injection

CIS-V8

- **5.4** — Restrict Administrator Privileges to Dedicated Administrator Accounts
- **6.8** — Define and Maintain Role-Based Access Control
- **16.10** — Apply Secure Design Principles in Application Architectures

ISO-27001-2022

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1190	Exploit Public-Facing Application	Initial-Access
T1546	Event Triggered Execution	Privilege-Escalation
T1059.004	Unix Shell	Execution

Sources

Source	URL	Tier
nvd	https://nvd.nist.gov/vuln/detail/CVE-2026-7465	T1
CVE-2026-7465 | Mondoo Vulnerability Intelligence	https://mondoo.com/vulnerability-intelligence/vulnerability/CVE-202...	T3
Website Builder for the Block Editor Vulnerability (CVE-2026-7465)	https://freshysites.com/security-bulletins/spectra-gutenberg-blocks...	T3
CVE-2026-5465 Detail - NVD	https://nvd.nist.gov/vuln/detail/CVE-2026-5465	T1
CVE-2026-7465 - CVE Record	https://www.cve.org/CVERecord?id=CVE-2026-7465	T3

Generated 2026-06-02 06:32 UTC by TJS Security Command Center