INTELLIGENCE BRIEFING
Security Command Center
TLP:CLEAR
2026-05-04 13:27 UTC

CVE-2026-2931: Amelia Booking Plugin for WordPress IDOR Allows Privilege Escalation to Admin Account Takeover

CVE VULNERABILITY | HIGH | CVSS 8.8 | CISA KEV

SCC Item ID	SCC-CVE-2026-0119
Type	CVE Vulnerability
CVE ID	CVE-2026-2931
Severity	HIGH
CVSS Base Score	8.8
EPSS Score	0.0005 (16th percentile)
KEV Status	Yes — CISA Known Exploited Vulnerability
Affected Products	tms-outsource Amelia Booking Plugin for WordPress, versions up to and including 9.1.2 (free and pro variants)
Published	2026-05-04T00:00:00Z
Discovery Source	Vulncheck Kev

Executive Summary

A high-severity vulnerability in the Amelia Booking plugin for WordPress allows any authenticated user with basic customer permissions to take over administrator accounts by changing arbitrary passwords through an insecure object reference flaw. Any WordPress site running Amelia versions through 9.1.2 is at risk of full site compromise. CISA has added this vulnerability to its Known Exploited Vulnerabilities catalog, confirming active exploitation in the wild.

Technical Analysis

CVE-2026-2931 is an Insecure Direct Object Reference (IDOR) vulnerability in the Amelia Booking Plugin for WordPress (vendor: tms-outsource), affecting all versions through 9.1.2 in both free and pro variants (both installed under the plugin slug ameliabooking). Classified under CWE-284 (Improper Access Control) and CWE-639 (Authorization Bypass Through User-Controlled Key), the flaw permits an authenticated attacker with customer-level access or higher to manipulate object references in API/form requests to modify passwords for arbitrary user accounts, including administrator accounts, without authorization checks. Successful exploitation yields full WordPress administrative access and complete site compromise. CVSS base score: 8.8 (High).

EPSS: 0.0053% (0.16th percentile at time of publication). CISA KEV listing confirms active exploitation. MITRE ATT&CK mappings: T1548 (Abuse Elevation Control Mechanism) and T1078 (Valid Accounts). Patch status: upgrade to version 9.1.3 or later. No CVSS vector string was included in source data.

Action Checklist

1. Step 1: Containment. Immediately identify all WordPress instances in your environment running the Amelia Booking plugin (free or pro, plugin slug 'ameliabooking'). If patching cannot happen within 24 hours, disable the plugin via the WordPress admin panel or rename the plugin directory on the server to prevent loading. For internet-facing sites, apply a WAF rule blocking POST requests to /wp-admin/admin-ajax.php with Amelia action parameters until patching is complete.
2. Step 2: Detection. Review WordPress access logs and application logs for anomalous POST requests to wp-admin/admin-ajax.php referencing Amelia action hooks, particularly password update or user modification actions originating from low-privilege authenticated sessions. Check wp_users and wp_usermeta tables in your WordPress database for unexpected changes to administrator account passwords or email addresses, especially recent modifications not tied to known administrative activity. Review authentication logs for new or unexpected admin-level logins.
3. Step 3: Eradication. Upgrade the Amelia Booking plugin to version 9.1.3 or later through the WordPress dashboard (Plugins > Installed Plugins > Update) or via WP-CLI: 'wp plugin update ameliabooking'. Confirm the installed version post-update. If exploitation is suspected, rotate all WordPress administrator account passwords immediately and audit user accounts for unauthorized additions or privilege changes.
4. Step 4: Recovery. After patching, validate that the installed plugin version is 9.1.3 or higher. Audit all WordPress administrator accounts: verify count, email addresses, and last login timestamps against known-good records. Re-enable the plugin only after confirming upgrade. Monitor authentication logs for continued anomalous admin logins for a minimum of 72 hours post-remediation. Verify site integrity by reviewing recently modified files and checking for injected backdoors or unauthorized plugin/theme additions.
5. Step 5: Post-Incident. This vulnerability exposes a control gap in plugin inventory management and privileged account monitoring. Implement a WordPress plugin inventory process with version tracking and patch SLAs. Enforce the principle of least privilege for WordPress user roles; audit whether customer-level registration on your booking sites is necessary and restrict if not. Deploy file integrity monitoring on WordPress installations. Add WordPress admin account changes as a monitored alert condition in your SIEM.

IR / Forensic Enrichment

Triage Priority	IMMEDIATE
Escalation Criteria	Escalate to CISO and legal counsel immediately if database queries reveal any WordPress administrator account password or email was modified by a non-admin session, or if Amelia customer booking records (containing PII such as names, email addresses, and phone numbers) were accessible post-exploitation, as CISA KEV listing combined with confirmed exploitation triggers breach notification assessment under applicable data protection regulations.

Recovery Notes	After patching to Amelia 9.1.3 and rotating all administrator credentials, validate site integrity by running 'wp plugin verify-checksums --all' and 'wp core verify-checksums' to confirm no attacker-planted files persist in the WordPress installation. Monitor Apache/Nginx access logs for renewed POST activity to '/wp-admin/admin-ajax.php' from customer-role sessions for a minimum of 72 hours, as threat actors who successfully exploited CVE-2026-2931 may have established persistent access via web shells or rogue administrator accounts before the patch was applied. If any unauthorized admin accounts or modified PHP files are discovered during recovery validation, treat the site as fully compromised and consider a clean rebuild from a known-good backup predating the earliest suspected exploitation timestamp.
Forensic Artifacts	Apache/Nginx access logs: POST requests to '/wp-admin/admin-ajax.php' with body parameters referencing Amelia action hooks (e.g., 'action=ameliaFrontendAjax' or similar Amelia-registered AJAX actions) paired with HTTP 200 responses and WordPress authentication cookies belonging to customer-role user IDs — this is the direct network trace of the IDOR exploit chain for CVE-2026-2931. | MySQL binary log (binlog) or direct wp_users table dump: SQL UPDATE statements targeting 'user_pass' or 'user_email' columns for rows where the associated wp_usermeta 'wp_capabilities' value contains 'administrator', with timestamps correlating to Amelia AJAX POST activity from non-admin sessions. | WordPress wp_usermeta table: rows where 'meta_key' is 'wp_capabilities' and 'meta_value' contains 'administrator' for user accounts not present in the pre-incident baseline, indicating attacker-created administrator accounts established after successful password takeover via CVE-2026-2931. | WordPress webroot recently modified PHP files: output of 'find /var/www/html/wp-content -name "*.php" -newer /var/www/html/wp-login.php -ls' identifying files created or modified after the earliest suspected exploitation timestamp, which may represent web shells or malicious plugins dropped by an attacker who leveraged the escalated administrator session for persistence. | Amelia plugin database tables (wp_amelia_customers, wp_amelia_appointments, wp_amelia_users): full export to assess whether attacker queries or modifications targeted customer PII stored by the booking plugin, which is a separate harm vector from the privilege escalation itself and directly relevant to breach notification scope.

Per-Action IR Details

Step 1: Containment — Immediately identify all WordPress instances in your environment running the Amelia Booking plugin (free or pro, plugin slug 'ameliabooking'). If patching cannot happen within 24 hours, disable the plugin via the WordPress admin panel or rename the plugin directory on the server to prevent loading. For internet-facing sites, apply a WAF rule to block unauthenticated and customer-role HTTP requests targeting Amelia's AJAX endpoints (/wp-admin/admin-ajax.php with Amelia action parameters), pending upgrade.

NIST Phase: Containment

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

Controls: NIST IR-4 (Incident Handling), NIST CM-7 (Least Functionality), CIS 4.4 (Implement and Manage a Firewall on Servers), CIS 2.3 (Address Unauthorized Software)

Compensating: Run 'find /var/www -type d -name ameliabooking' across all web roots to enumerate affected instances, then execute 'mv /path/to/wp-content/plugins/ameliabooking /path/to/wp-content/plugins/ameliabooking.DISABLED' to break plugin load without touching the database. For WAF-less environments, add an Apache/Nginx rule to return 403 on POST requests to '/wp-admin/admin-ajax.php' where the body contains 'amelia' (case-insensitive match on action parameter); example Nginx snippet: 'if ($request_body ~* "amelia") { return 403; }' under the admin-ajax location block.

Evidence: Before disabling the plugin, snapshot the wp_users and wp_usermeta database tables ('mysqldump -u root -p wordpress wp_users wp_usermeta > pre_containment_snapshot_$(date +%F).sql') to preserve the pre-containment state of all account records including user_pass hashes, user_email, and meta_value entries for 'wp_capabilities'. Also

capture a directory listing with timestamps ('ls -laR /var/www/html/wp-content/plugins/ameliabooking/ > plugin_file_inventory.txt') to establish whether plugin files were modified post-installation, which may indicate a secondary web shell drop following exploitation.

Step 2: Detection — Review WordPress access logs and application logs for anomalous POST requests to wp-admin/admin-ajax.php referencing Amelia action hooks, particularly password update or user modification actions originating from low-privilege authenticated sessions. Check wp_users and wp_usermeta tables in your WordPress database for unexpected changes to administrator account passwords or email addresses, especially recent modifications not tied to known administrative activity. Review authentication logs for new or unexpected admin-level logins. No public IOC signatures were available in the source data at time of writing.

NIST Phase: Detection Analysis

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

Controls: NIST AU-6 (Audit Record Review, Analysis, and Reporting), NIST AU-3 (Content of Audit Records), NIST SI-4 (System Monitoring), CIS 8.2 (Collect Audit Logs)

Compensating: Execute the following against the Apache/Nginx access log to isolate Amelia IDOR exploit attempts: 'grep -E "POST.*admin-ajax\.php" /var/log/apache2/access.log | grep -i "amelia" | awk '{print $1, $4, $7, $9}' | sort | uniq -c | sort -rn'. Then query the WordPress database directly to identify password or email changes on administrator accounts not performed by a known admin session: 'mysql -u root -p -e "SELECT ID, user_login, user_email, user_registered, user_pass FROM wp_users WHERE ID IN (SELECT user_id FROM wp_usermeta WHERE meta_key=\"wp_capabilities\" AND meta_value LIKE \"%administrator%\") ORDER BY user_pass;"'. Cross-reference the timestamps of any changes against the POST log entries from the same source IP.

Evidence: Collect: (1) Full Apache/Nginx access logs covering the Amelia plugin install date through present, specifically POST entries to '/wp-admin/admin-ajax.php' with HTTP 200 responses from non-admin authenticated sessions — the IDOR exploitation of CVE-2026-2931 requires an authenticated customer-role session, so look for session cookies or WordPress auth cookies paired with those requests. (2) WordPress debug log ('/wp-content/debug.log' if WP_DEBUG_LOG is enabled) for Amelia function calls to password update or user object modification hooks. (3) Database binary log (if enabled) to reconstruct the exact SQL UPDATE statements issued against wp_users.user_pass and wp_users.user_email for administrator-role accounts.

Step 3: Eradication — Upgrade the Amelia Booking plugin to version 9.1.3 or later through the WordPress dashboard (Plugins > Installed Plugins > Update) or via WP-CLI: 'wp plugin update ameliabooking'. Confirm the installed version post-update. If exploitation is suspected, rotate all WordPress administrator account passwords immediately and audit user accounts for unauthorized additions or privilege changes.

NIST Phase: Eradication

Reference: NIST 800-61r3 §3.4 — Eradication

Controls: NIST SI-2 (Flaw Remediation), NIST IA-5 (Authenticator Management), NIST AC-2 (Account Management), CIS 7.4 (Perform Automated Application Patch Management), CIS 5.1 (Establish and Maintain an Inventory of Accounts)

Compensating: Update via WP-CLI with version confirmation: 'wp plugin update ameliabooking --version=9.1.3 && wp plugin get ameliabooking --field=version'. Immediately rotate all administrator passwords using WP-CLI: 'wp user list --role=administrator --field=ID | xargs -I{} wp user update {} --user_pass=$(openssl rand -base64 20)' and force logout of all active sessions by regenerating WordPress auth keys in wp-config.php (replace AUTH_KEY, SECURE_AUTH_KEY, LOGGED_IN_KEY, and NONCE_KEY with new values from 'https://api.wordpress.org/secret-key/1.1/salt/'). Audit for unauthorized administrator accounts: 'wp user list --role=administrator --fields=ID,user_login,user_email,user_registered'.

Evidence: Before applying the patch, preserve: (1) A copy of the vulnerable Amelia plugin files ('tar -czf ameliabooking_pre_patch_$(date +%F).tar.gz /var/www/html/wp-content/plugins/ameliabooking/') as forensic evidence of the vulnerable codebase, specifically the AJAX handler functions responsible for the IDOR flaw (look in 'src/Application/Commands/User/' or similar path for password update command handlers). (2) A full dump of wp_users and wp_usermeta showing current user_pass hashes and wp_capabilities values for all accounts — this

establishes the post-exploitation state before remediation overwrites attacker-modified password hashes.

Step 4: Recovery — After patching, validate that the installed plugin version is 9.1.3 or higher. Audit all WordPress administrator accounts: verify count, email addresses, and last login timestamps against known-good records. Re-enable the plugin only after confirming upgrade. Monitor authentication logs for continued anomalous admin logins for a minimum of 72 hours post-remediation. Verify site integrity by reviewing recently modified files and checking for injected backdoors or unauthorized plugin/theme additions.

NIST Phase: Recovery

Reference: NIST 800-61r3 §3.5 — Recovery

Controls: NIST IR-4 (Incident Handling), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AU-6 (Audit Record Review, Analysis, and Reporting), CIS 7.2 (Establish and Maintain a Remediation Process), CIS 5.3 (Disable Dormant Accounts)

Compensating: Run 'find /var/www/html -name "*.php" -newer /var/www/html/wp-config.php -mtime -30 -ls > recently_modified_php.txt' to surface PHP files modified within the last 30 days that postdate the wp-config.php baseline — newly planted web shells or backdoor plugins will appear here. Validate plugin integrity using WP-CLI: 'wp plugin verify-checksums ameliabooking' and 'wp core verify-checksums'. For admin account validation without a SIEM, query: 'wp user list --role=administrator --fields=ID,user_login,user_email,user_registered,last_login' and diff against a known-good baseline captured before the incident. Schedule an hourly cron job for 72 hours to append new admin logins to a watch file: 'grep "wp-login.php" /var/log/apache2/access.log | grep " 200 " >> /var/log/admin_login_watch.log'.

Evidence: Collect before re-enabling the plugin: (1) Output of 'wp plugin verify-checksums ameliabooking' confirming 9.1.3 checksum integrity against the WordPress.org repository — document the command output as evidence of verified eradication. (2) A final snapshot of wp_users and wp_usermeta post-password-rotation to confirm no residual attacker-controlled administrator accounts persist. (3) Results of the recently-modified PHP file scan to confirm absence of web shells or backdoors that an attacker who successfully exploited CVE-2026-2931 (gaining admin credentials) may have dropped to maintain persistence after patching.

Step 5: Post-Incident — This vulnerability exposes a control gap in plugin inventory management and privileged account monitoring. Implement a WordPress plugin inventory process with version tracking and patch SLAs. Enforce the principle of least privilege for WordPress user roles — audit whether customer-level registration on your booking sites is necessary and restrict if not. Deploy file integrity monitoring on WordPress installations. Add WordPress admin account changes as a monitored alert condition in your SIEM.

NIST Phase: Post Incident

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

Controls: NIST IR-5 (Incident Monitoring), NIST IR-8 (Incident Response Plan), NIST SI-7 (Software, Firmware, and Information Integrity), NIST AC-6 (Least Privilege), NIST CM-8 (System Component Inventory), CIS 1.1 (Establish and Maintain Detailed Enterprise Asset Inventory), CIS 2.1 (Establish and Maintain a Software Inventory), CIS 7.1 (Establish and Maintain a Vulnerability Management Process), CIS 5.4 (Restrict Administrator Privileges to Dedicated Administrator Accounts)

Compensating: Deploy the free WP Crontrol or WP Activity Log plugin to emit syslog-format alerts on WordPress administrator account modifications (user_pass or user_email changes, capability promotions), which can be ingested by any syslog receiver or written to a monitored log file. For file integrity monitoring without a commercial tool, configure AIDE (Advanced Intrusion Detection Environment) or Tripwire Open Source against the WordPress webroot with a weekly cron baseline scan and email diff report. Create a WP-CLI-based plugin inventory script ('wp plugin list --fields=name,version,status --format=csv > plugin_inventory_$(date +%F).csv') run weekly and diffed against the prior week to catch unauthorized plugin additions or version regressions — this directly addresses the control gap exposed by CVE-2026-2931.

Evidence: Document the following for the post-incident lessons-learned record: (1) Timeline reconstruction from access logs showing the earliest POST to '/wp-admin/admin-ajax.php' with an Amelia action parameter from a customer-role session — this establishes the likely exploitation window and informs the breach notification assessment

if administrator credentials or customer PII stored in the Amelia booking database were accessed. (2) Full list of administrator accounts present at time of discovery versus known-good baseline, documenting any accounts created or modified by the attacker. (3) Evidence of whether the Amelia booking database tables (wp_amelia_customers, wp_amelia_appointments) containing customer PII were accessible to the attacker post-privilege-escalation, as this determination drives regulatory notification obligations.

Detection Guidance

Query web server access logs for high-frequency POST requests to /wp-admin/admin-ajax.php from authenticated low-privilege sessions, particularly those containing Amelia-specific action parameters related to user or password updates. In the WordPress database, run: SELECT user_login, user_pass, user_registered FROM wp_users WHERE user_registered > [date of concern] OR ID IN (SELECT user_id FROM wp_usermeta WHERE meta_key='wp_capabilities' AND meta_value LIKE '%administrator%'); compare results against your known administrator roster. Monitor for WordPress admin logins from new IP addresses or outside normal hours. SIEM alert: flag any wp_users password field change event not correlated with a known admin-initiated reset workflow. Note: No confirmed public exploit signatures or specific IOC patterns were available in the source data at time of analysis.

Framework Mappings

MITRE-ATTACK

- **T1548** — Abuse Elevation Control Mechanism
- **T1078** — Valid Accounts

NIST-800-53R5

- **AC-6** — Least Privilege
- **CM-6** — Configuration Settings
- **AC-2** — Account Management
- **IA-2** — Identification and Authentication (Organizational Users)
- **IA-5** — Authenticator Management
- **AC-3** — Access Enforcement

OWASP-TOP10-2021

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

CIS-V8

- **6.1** — Establish an Access Granting Process
- **6.2** — Establish an Access Revoking Process
- **5.4** — Restrict Administrator Privileges to Dedicated Administrator Accounts

SOC2-TSC

- **CC6.1** — The entity implements logical access security software, infrastructure, and architectures over protected information assets
- **CC6.3** — Authorizes, modifies, or removes access

HIPAA-SECURITY

- **164.312(a)(1)** — Access Control

ISO-27001-2022

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

MITRE ATT&CK Mapping

Technique ID	Technique Name	Tactic
T1548	Abuse Elevation Control Mechanism	Privilege-Escalation
T1078	Valid Accounts	Defense-Evasion

Sources

Source	URL	Tier
vulncheck_kev	https://nvd.nist.gov/vuln/detail/CVE-2026-2931	T1
CVE-2026-2931 - Red Hat Customer Portal	https://access.redhat.com/security/cve/cve-2026-2931	T3
CVE-2026-2931 - Exploits & Severity - Feedly	https://feedly.com/cve/CVE-2026-2931	T3
CVE-2026-2931 - CVE Record	https://www.cve.org/CVERecord?id=CVE-2026-2931	T3
CVE-2026-2931: Amelia Booking Auth Bypass Vulnerability	https://www.sentinelone.com/vulnerability-database/cve-2026-2931/	T3
CISA KEV	https://www.cisa.gov/known-exploited-vulnerabilities-catalog	T1

Generated 2026-05-04 13:27 UTC by TJS Security Command Center