SECURITY ASSESSMENT REPORT

Web Application — SEO Injection, Credential Exposure & Full Remediation

---

Prepared for: [Client] · Confidential

Analyst: Daniel Oa · danieloa.com

Date: May 2026 · Classification: CONFIDENTIAL

1. Executive Summary

A comprehensive security assessment of a client WordPress/WooCommerce web application revealed an active compromise of the production environment. A threat actor had injected hidden SEO spam content directly into the CMS database, exploiting the client's domain authority to rank illegal gambling and betting websites on Google Search — a technique known as **Hidden Text SEO Injection**. The attack was fully operational and had persisted undetected for at least 2.5 months across three distinct injection waves.

The engagement covered both phases of the incident response: **discovery and forensic analysis** (audit) followed by **complete technical remediation** executed the following day. All critical and high-severity findings were fully resolved without disruption to live site functionality.

5	3	1	8
CRITICAL	HIGH	MEDIUM	RESOLVED

2. Scope & Methodology

Target	PHP/WordPress 6.7 + WooCommerce + Elementor Pro web application (production server)
Testing Type	Black-box — file system access via FTP-TLS + full database dump forensic analysis
Methods	Static file analysis · SQL dump forensics · Database inspection · FTP file review · Code review
Framework	OWASP Testing Guide v4.2
Audit Date	May 5, 2026
Remediation Date	May 6, 2026
Analyst	Daniel Oa — danieloa.com

3. Findings Summary

ID	Title	Severity	CVSS	Status
F-01	SEO Spam injection in CMS database (Hidden Text — 60 hidden divs + 1px widget)	Critical	9.5	Resolved
F-02	Full site backup (3.8 GB) publicly accessible without authentication	Critical	9.8	Resolved
F-03	WordPress admin credentials hardcoded in PHP source files (Base64-encoded)	Critical	9.1	Resolved
F-04	Customer PII log file (names, phones, addresses) accessible via HTTP	Critical	8.2	Resolved
F-05	Development directory with internal scripts and webhooks publicly accessible	Critical	9.1	Resolved
F-06	Six WordPress users with Administrator role — least privilege principle violated	High	5.4	Resolved
F-07	REST API exposes admin usernames without authentication	High	6.5	Resolved
F-08	All administrator passwords compromised / matching publicly exposed credentials	High	8.0	Resolved
F-09	GET parameters in custom PHP scripts injected into API calls without sanitization (XSS)	Medium	6.1	Pending

4. Detailed Findings

F-01 — SEO Spam Injection in CMS Database	CRITICAL · CVSS 9.5
Description	Hidden HTML content was injected into the `_elementor_data` field of the WordPress database across 25 records. Two injection techniques were used simultaneously: (1) 60 `<div>` blocks with `position:absolute` and large negative coordinates (left:-8713px, top:-7864px) — completely invisible to human visitors but fully indexed by Google crawlers; (2) one Elementor HTML widget with white text at 1px font-size. Both techniques served links to 51 illegal gambling and betting domains in 5 languages (Italian, Polish, Spanish, Portuguese, German). Three distinct injection waves were identified between February 17 and April 29, 2026, confirming persistent attacker access for at least 2.5 months.
Impact	The client's domain authority was actively exploited to rank illegal content on Google Search. Risk of permanent domain blacklisting, loss of all organic search traffic, and significant reputational damage. Attack was invisible to all browser-based monitoring tools.
Evidence	Database table `wp_postmeta`, field `_elementor_data` (homepage record). Multiple infected post revisions identified. 51 unique gambling domains in injected content. Significant reduction in field size confirmed post-cleanup.
Remediation	SQL script executed via database management interface: deleted all infected revisions from `wp_posts`, `wp_postmeta`, and `wp_term_relationships`. Cleaned homepage `_elementor_data` with precision regex targeting both injection patterns. Elementor cache cleared. All page sections and legitimate widgets preserved intact. Site verified operational post-cleanup.

F-02 — Full Site Backup Publicly Accessible	CRITICAL · CVSS 9.8
Description	A 3.8 GB backup archive generated by the All-in-One WP Migration plugin was accessible via HTTP without any authentication. The archive contains the complete site source code, database dump with all customer data, and `wp-config.php` with MySQL credentials and WordPress secret keys.
Impact	Any attacker can download the full environment and extract all credentials, customer PII, order history, and configuration secrets. This is identified as the most probable initial attack vector for the SEO spam compromise.
Remediation	Added `RedirectMatch 403` directive to `wp-content/.htaccess` (parent directory level) to block all HTTP access to `.wpress` files. Parent-level placement prevents the plugin from overwriting the protection when generating new backups.

F-03 — Admin Credentials Hardcoded in PHP Source	CRITICAL · CVSS 9.1
Description	WordPress administrator credentials were hardcoded as a Base64-encoded HTTP Basic Authentication token in 5 PHP files within a publicly accessible development directory. The token decoded to plaintext credentials and appeared in 13 locations across the codebase.
Impact	Any visitor who accessed these files could extract administrator credentials with a single Base64 decode operation. Combined with the accessible backup (F-02), credentials were available via two independent vectors.
Remediation	Created centralized `dev/config.php` with credential as a PHP constant (`API_AUTH_TOKEN`). Blocked HTTP access to `config.php` via `dev/.htaccess` . Updated all 5 PHP scripts to reference the constant via `require_once` . After admin password rotation, `config.php` was immediately updated with the new token to maintain uninterrupted service.

F-04 — Customer PII Accessible via HTTP	CRITICAL · CVSS 8.2
Description	A log file written by the WooCommerce order webhook contained real customer data including full names, phone numbers, delivery addresses, and order amounts. The file was accessible via direct HTTP request with no authentication.
Impact	Exposure of customer Personally Identifiable Information (PII) constitutes a data breach under applicable data protection legislation. Risk of regulatory fines and reputational damage.
Remediation	Created `dev/[integration]/webhook/.htaccess` blocking HTTP access to all `.txt` files. The webhook PHP script retains filesystem write access to the log — only direct HTTP reads are blocked. Access verified denied post-deployment.

F-05 — Development Directory Publicly Exposed	HIGH · CVSS 9.1
Description	The `/dev/` directory containing internal ERP synchronization scripts, webhook handlers, product configurator integrations, and sensitive log files was publicly accessible without any access restriction.
Remediation	Selective protection applied per sub-path after functional analysis: public configurators (`/dev/configurator-*/`) must remain accessible as they serve live site visitors; webhook handler (`/dev/[integration]/webhook/`) protected at file-type level (logs blocked, PHP accessible). Credential file protected at directory level. ERP sync tools left accessible as they require manual browser-triggered operation by internal staff.

F-06 — Six Administrator Accounts — Least Privilege Violation	HIGH · CVSS 5.4
Description	All 6 WordPress users held the Administrator role. Users with content-only responsibilities (marketing, operations) had unnecessary access to install plugins, modify themes, and alter site configuration.
Remediation	Role audit performed. 3 users confirmed as requiring Administrator access. 3 users downgraded to Editor via `wp_usermeta` SQL update. Editors retain full content creation capabilities but cannot modify system configuration.

F-07 — REST API User Enumeration	HIGH · CVSS 6.5
Description	The WordPress REST API endpoint `/wp-json/wp/v2/users` responded with a full list of administrator usernames, display names, and partial emails without requiring any authentication — enabling targeted brute force attacks.
Remediation	Added `rest_endpoints` filter to active theme's `functions.php`, removing the `/wp/v2/users` and `/wp/v2/users/{id}` endpoints from the REST API registry. Verified no application scripts depended on this endpoint prior to implementation.

F-08 — Administrator Passwords Compromised	HIGH · CVSS 8.0
Description	The primary administrator password was exposed in source code (F-03) and included in the publicly accessible site backup (F-02). Hosting provider security logs recorded hundreds of unauthorized access attempts to the WordPress login page. All administrator accounts used passwords of unknown exposure status.
Remediation	All administrator passwords rotated simultaneously via `wp_users` SQL update. Strong 16-character passwords generated with mixed character classes. `config.php` updated with new credential token within minutes of password rotation to preserve uninterrupted live site functionality.

5. Remediation Verification

A post-remediation test suite of 8 automated checks was executed immediately after completing all changes. All tests passed. No live site functionality was disrupted.

#	Test	Result
T-01	Backup .wpress file returns HTTP 403 — `wp-content/.htaccess` rule active	PASS
T-02	PII log file returns HTTP 403 — `webhook/.htaccess` rule active	PASS
T-03	`config.php` returns HTTP 403 — `dev/.htaccess` block active	PASS
T-04	Token in `config.php` decodes to updated administrator credentials	PASS
T-05	No PHP script contains the previously exposed Base64 credential token	PASS
T-06	`rest_endpoints` filter present in theme `functions.php`	PASS
T-07	WooCommerce webhook PHP handler remains accessible (not blocked)	PASS
T-08	REST API `/wp/v2/users` returns no user data	PASS

> All changes were validated against the live application before deployment. The product configurators, order webhook, and ERP synchronization tools continued operating without interruption throughout the remediation process.

6. Remediation Roadmap

Items below remain open. All critical and high-severity findings have been resolved.

Phase	Finding	Action
Short-term (< 1 week)	F-09	Sanitize `$_GET` parameters in custom PHP scripts using `sanitize_text_field()` or `htmlspecialchars()`
Short-term (< 1 week)	—	Enable 2FA on all administrator accounts (WP 2FA or Wordfence)
Medium-term (< 1 month)	—	Delete 3.8 GB backup file from server (currently blocked via HTTP; file still occupies disk)
Medium-term (< 1 month)	—	Submit reconsideration request to Google Search Console after confirming clean index
Medium-term (< 1 month)	—	Implement WAF (Cloudflare proxy or Sucuri) for ongoing traffic filtering
Ongoing	—	Follow-up security audit 30 days post-remediation · Subscribe to WPScan/ Patchstack CVE alerts · File integrity monitoring via Sucuri or Wordfence

7. Disclaimer

This assessment was conducted via black-box analysis including FTP-TLS file system inspection, SQL database dump forensics, and static code review. It does not constitute a full penetration test. Dynamic exploitation, authenticated web application testing, and network-layer assessment are outside the scope of this engagement. All findings were responsibly disclosed to the client and remediation was executed with explicit client authorization. This report is confidential and intended solely for the authorized recipient.