SECURITY ASSESSMENT REPORT

Web Application — SEO Injection & Malware Discovery

Prepared for: [Client] · Confidential

Analyst: Daniel Oa · danieloa.com

Date: April 2026 · Classification: CONFIDENTIAL

1. Executive Summary

A security assessment of a client web application revealed an active server-side attack that had compromised the production environment without the client's knowledge. Malicious PHP code had been planted on the web server to serve gambling and illegal content to Google's search crawlers while displaying the legitimate site to all human visitors — a technique known as **SEO Cloaking**. The attack exploited the client's trusted domain authority to rank illicit content on Google Search, placing the domain at risk of permanent blacklisting and significant reputational damage. Ten vulnerabilities were identified across application and server layers through manual code review, file system inspection, and database analysis.

CRITICAL	HIGH	MEDIUM	LOW
1	3	4	2

2. Scope & Methodology

Target	PHP/WordPress web application (production server)
Testing Type	Black-box — no source code provided prior to assessment
Methods	Manual code review · File system analysis · Database inspection · Log review
Framework	OWASP Testing Guide v4.2
Period	April 2026
Analyst	Daniel Oa — danieloa.com

3. Findings Summary

ID	Title	Severity	CVSS
F-01	Malicious PHP backdoor — SEO cloaking	Critical	9.1
F-02	Remote content injection via cURL	High	8.8
F-03	Unrestricted file write on web root	High	8.5
F-04	Compromised CMS administrator credentials	High	8.0
F-05	Outdated PHP / WordPress / plugins	High	7.5
F-06	No file integrity monitoring	Medium	6.5
F-07	DB credentials in web-accessible path	Medium	6.0
F-08	SQL injection via vulnerable plugin layer	Medium	5.8
F-09	Missing security headers (CSP/HSTS/XFO)	Medium	4.3
F-10	Directory listing on subfolders	Low	3.7

4. Detailed Findings

F-01 — Malicious PHP Backdoor — SEO Cloaking — CRITICAL · CVSS 9.1

Description	A PHP file was planted in a web-accessible directory to replace the legitimate index.php. It implements a two-step Google crawler verification: (1) IP matching against hardcoded Google CIDR blocks and (2) forward-confirmed reverse DNS (FCrDNS). Once verified, the backdoor fetches gambling and illegal content from an attacker-controlled Cloudflare Pages subdomain and serves it directly to Googlebot. Human visitors receive the legitimate site, making the attack invisible to all browser-based monitoring.
Impact	The client's domain authority is being exploited to rank illegal content on Google Search. Risk of permanent domain blacklisting, loss of organic traffic, and reputational damage. The attack was fully operational and undetected on the production server.
Remediation	1. Immediately remove the malicious file and restore from a verified clean backup. 2. Rotate all CMS and server credentials. 3. Deploy file integrity monitoring to detect future unauthorized changes. 4. Submit a reconsideration request to Google Search Console after cleanup.

F-02 — Remote Content Injection via cURL — HIGH · CVSS 8.8

Description	The malicious backdoor uses PHP's cURL library to fetch arbitrary HTML content from an external attacker-controlled domain (Cloudflare Pages) and serve it under the victim's domain.
Impact	An attacker can serve any content — malware, phishing, or illegal material — under the client's trusted domain. The content can be changed at any time without touching the server.
Remediation	1. Disable outbound cURL in php.ini: disable_functions = curl_exec, curl_init 2. Implement outbound egress firewall rules to block unauthorized external connections. 3. Audit all PHP files for cURL usage.

F-03 — Unrestricted File Write on Web Root — HIGH · CVSS 8.5

Description	The attacker was able to write and execute a PHP file directly on the web root, indicating a lack of file permission controls or an exploited write path (likely via a compromised CMS credential or vulnerable plugin).
Impact	Persistent server-side code execution. Any file written to the web root by an attacker is immediately executable as PHP.
Remediation	1. Restrict web root write permissions: chmod 755 on directories, 644 on files. 2. Disable PHP execution in upload directories via .htaccess. 3. Run a full file audit to identify any other unauthorized files.

F-04 — Compromised CMS Administrator Credentials — HIGH · CVSS 8.0

Description	Evidence in the malicious file and server logs indicates the attacker gained access using valid CMS administrator credentials, either through credential stuffing, brute force, or a previously leaked password.
Impact	Full CMS takeover. An attacker with admin credentials can install plugins, modify themes, create backdoors, and access all site content and user data.
Remediation	1. Immediately reset all CMS administrator passwords. 2. Enable multi-factor authentication on all admin accounts. 3. Audit the admin user list and remove any unknown accounts. 4. Review login logs for unauthorized access history.

F-05 — Outdated PHP / WordPress Core / Plugins

HIGH · CVSS 7.5

Description	The server runs outdated versions of PHP, WordPress core, and several plugins, including components with publicly known CVEs (CVE-2024-6386, CVE-2024-8858, CVE-2024-9079).
Impact	Known exploits exist for the installed versions. Remote code execution and privilege escalation attacks are possible using publicly available tools.
Remediation	1. Update PHP to the latest stable release (8.3+). 2. Update WordPress core to the latest version. 3. Update all plugins and themes — remove unused ones entirely. 4. Subscribe to WPScan or Patchstack for ongoing vulnerability alerts.

F-06 — No File Integrity Monitoring

MEDIUM · CVSS 6.5

Description	No file integrity monitoring system was in place. The malicious PHP file remained on the server undetected — the attack operated for an unknown period before discovery.
Impact	Future attacks can persist for extended periods without detection. There is no alerting mechanism for unauthorized file changes on the server.
Remediation	1. Deploy AIDE or Tripwire for server-level file integrity monitoring. 2. Enable Wordfence or Sucuri file change alerts within WordPress. 3. Configure real-time email/SMS alerts on any modification to web root files.

F-07 — Database Credentials in Web-Accessible Path

MEDIUM · CVSS 6.0

Description	The database hostname, username, and password are stored in wp-config.php within the web-accessible directory. While standard for WordPress, the file was present in a backup that was accessible.
Impact	If the backup or wp-config.php is exposed (via path traversal, misconfiguration, or server compromise), an attacker gains direct database access.
Remediation	1. Move wp-config.php one directory above the web root. 2. Restrict file permissions: chmod 600 wp-config.php 3. Add .htaccess protection: Require all denied

F-08 — SQL Injection via Vulnerable Plugin Layer

MEDIUM · CVSS 5.8

Description	One or more installed plugins contain known SQL injection vulnerabilities in their database query handling, allowing unsanitized user input to reach raw SQL queries.
Impact	An attacker can extract the full database contents including user credentials, private content, and personal data. In some configurations, file write via SQL injection is possible.
Remediation	1. Update or replace the affected plugins. 2. Enforce parameterized queries (prepared statements) in all custom queries. 3. Enable a WAF (Wordfence, Cloudflare) with SQL injection rules.

F-09 — Missing HTTP Security Headers

MEDIUM · CVSS 4.3

Description	The server does not send Content-Security-Policy (CSP), HTTP Strict Transport Security (HSTS), X-Frame-Options, or X-Content-Type-Options headers on any response.
Impact	Missing CSP amplifies the impact of any XSS vulnerability. Missing HSTS allows downgrade attacks. Missing X-Frame-Options enables clickjacking on all site pages.
Remediation	Add to .htaccess or server config: Header set X-Frame-Options SAMEORIGIN Header set X-Content-Type-Options nosniff Header set Strict-Transport-Security max-age=31536000 Header set Content-Security-Policy default-src 'self'

F-10 — Directory Listing Enabled on Subfolders

LOW · CVSS 3.7

Description	Several subdirectories expose their full file listing when accessed directly in a browser, revealing plugin structure, theme files, custom code, and upload paths.
Impact	Attackers can enumerate all files on the server to identify targets for further exploitation and understand the full application structure.
Remediation	Add to .htaccess: Options -Indexes Or in Nginx: autoindex off;

5. Remediation Roadmap

Prioritized remediation plan. Complete Immediate actions before proceeding to Short-term.

Phase	Finding	Action
Immediate (< 24h)	F-01, F-04	Remove malicious files · Rotate ALL credentials · Audit admin users
Short-term (< 1 week)	F-02, F-03, F-05	Patch PHP/WordPress/plugins · Disable cURL outbound · Fix file permissions
Medium-term (< 1 month)	F-06–F-09	Deploy FIM · Move wp-config · Add security headers · Patch SQL injection
Ongoing	F-10	Disable directory listing · Subscribe to CVE advisories · Monthly audits

6. Disclaimer

This assessment was conducted via manual black-box analysis including code review, file system inspection, and database analysis. It does not constitute a full penetration test. Dynamic exploitation, authenticated testing, and network-layer assessment are outside the scope of this engagement. All findings were responsibly disclosed to the client. This report is confidential and intended solely for the authorized recipient.