# WordPress Malware Incident Report

*Full Incident Lifecycle: Detection → Remediation → Verification*

**borderlessmigration.com.au**

Report Date: July 02, 2026

| CRITICAL | HIGH | MEDIUM | LOW | STATUS |
|---|---|---|---|---|
| 2 | 2 | 1 | 1 | CLEAN |

| | |
|---|---|
| **Site** | borderlessmigration.com.au |
| **Hosting** | cPanel (borderlessmigrat user) — Cloudflare proxied |
| **CMS** | WordPress — Theme: Dumketo — 12 active plugins |
| **Scope** | Database export (SQL) + full wp-content directory |
| **Initial scan** | 2026-07-02 — database analysis |
| **Remediation** | 2026-07-02 — manual cleanup by site owner via cPanel |
| **Verification** | 2026-07-02 — cleaned DB + wp-content re-scanned; Wordfence scan: 0 issues |
| **Analyst** | MD Emmon Hossain — WordPress & Malware Security Analyst |

## 1. Executive Summary

This report documents a complete WordPress malware incident for **borderlessmigration.com.au** — from initial detection through to confirmed remediation. The site was found to be actively compromised with multiple concurrent threats: a PHP database backdoor, a rogue administrator account with credentials already exfiltrated to a remote attacker, and obfuscated JavaScript injected directly into the active theme's footer template.

Four distinct malware components were identified and removed. Following cleanup, an independent verification scan of the database (19 MB SQL export) and the full wp-content directory tree (8,347 PHP files) confirmed zero remaining threats. A Wordfence scan of the live environment independently corroborated this result — 15,624 files scanned, **0 issues found**. The site is clean.

## 2. Findings — All Malware Identified

| # | Severity | Finding | Location | Type |
|---|---|---|---|---|
| F1 | CRITICAL | GAnalytics PHP Backdoor in Code Snippets | wp_snippets, row ID 5 | Backdoor / Persistence |
| F2 | CRITICAL | Rogue Administrator Account | wp_users ID 3, wp_usermeta | Unauthorised Access |
| F3 | HIGH | Active C2 Communication — credentials exfiltrated | wp_options (_transient___ga_r_cache) | Data Exfiltration |
| F4 | HIGH | Obfuscated JavaScript in theme footer.php | wp-content/themes/dumketo/footer.php line 24 | Theme Injection / Client-Side Attack |
| F5 | MEDIUM | Spam bot submissions in Flamingo log | wp_posts (flamingo_inbound) | Spam / Cleanup |
| F6 | LOW | create_function() in Dumketo theme widgets | themes/dumketo/inc/widget/ | Deprecated PHP code |

# 3. Detailed Findings

## F1 — GAnalytics PHP Backdoor (wp_snippets ID 5)

| CRITICAL | Database — wp_snippets row 5 — active = 1 |
|---|---|

A complete PHP backdoor class was injected into the Code Snippets plugin database table (wp_snippets) with the deceptive name *"Analytics Configuration"*. It was in the **active (running) state** (active = 1, modified 2026-07-01 17:45:37 UTC). The identifying marker at the head of the payload:

```
/*ed247e0ac47a37e2*/ — class GAwp_d1c0b780
```

This is a known fingerprint of the **GAnalytics / WP-Snippets-Backdoor** malware family (documented by Wordfence and Sucuri). The class hooked into nine WordPress actions/filters to: silently create a hidden admin account on every page load, filter that user from the admin panel and REST API, hide itself from the Code Snippets and WPCode plugin interfaces, block author archive pages, and load remote assets. It also actively self-concealed from the all_plugins filter so it would not appear in the plugin list.

| Hook | Method | Effect |
|---|---|---|
| init | createuser() | Creates hidden admin account on every page load |
| pre_user_query | filterusers() | Removes rogue user from admin Users list |
| rest_prepare_user | filter_rest_user() | Hides rogue user from REST API /wp/v2/users |
| pre_get_posts | block_author_archive() | Blocks author archive for rogue user |
| wp_sitemaps_users_query_args | filter_sitemap_users() | Hides rogue user from XML sitemaps |
| code_snippets/list_table/get_snippets | hide_from_code_snippets() | Hides itself from Code Snippets UI |
| wpcode_code_snippets_table_... | hide_from_wpcode() | Hides itself from WPCode UI |
| all_plugins | hplugin() | Hides itself from the plugin list |
| wp_enqueue_scripts | loadassets() | Loads remote scripts from C2 endpoint |

## F2 — Rogue Administrator Account (mail_daemon6fb98e18)

| CRITICAL | wp_users ID 3 / wp_usermeta (13 rows) |
|---|---|

The backdoor (F1) automatically created a hidden WordPress administrator on its first execution. The account was confirmed present in the initial database export:

| Field | Value |
|---|---|
| user_login | mail_daemon6fb98e18 |
| user ID | 3 |
| user_email | mail-daemon@www.borderlessmigration.com.au |

| | |
|---|---|
| user_registered | 2026-07-01 17:45:40 UTC — 3 seconds after the snippet was inserted |
| wp_capabilities | administrator (level 10 — full privileges) |
| Hidden via | wp_options: __ga_hidden_users = ["mail_daemon6fb98e18"] |

The username and password were derived deterministically from the site's DB_PASSWORD and AUTH_SALT constants, allowing the attacker to compute valid credentials for any site running this backdoor. Credentials were immediately transmitted to the C2 server (see F3).

### F3 — Active C2 Communication & Credential Exfiltration

| HIGH | wp_options: _transient___ga_r_cache |
|---|---|

The backdoor resolved a command-and-control (C2) endpoint by querying a shuffled pool of 18 attacker-controlled domains and caching the result. At the time of the database export the active C2 server was:

```
https://waterpump41.world (cached in _transient___ga_r_cache)
```

The 18 resolver domains embedded in the backdoor spanned multiple registrars and TLDs to ensure resilience: *metricaxiom.icu/.live, neuralprobe.mobi, synthquant.info, datumflux.fit/.ink/.art, vanguardcogni.sbs/.pro/.icu/.shop/.xyz, nexusquant.top/.info/.shop/.icu/.live/.pro.* After creating the rogue admin account, the backdoor immediately POSTed the site domain, a public key, the generated username, and the generated password to /api/sites/setup-credentials on the resolved C2 endpoint. The attacker therefore already held valid administrator credentials for this site at the time of discovery.

### F4 — Obfuscated JavaScript Injected into footer.php

| HIGH | wp-content/themes/dumketo/footer.php — line 24 |
|---|---|

During manual file review via cPanel File Manager, a large block of heavily obfuscated JavaScript was discovered injected directly into the active theme's **footer.php** template, beginning at line 24. The injection appeared immediately after the legitimate wp_footer() call and before the closing <?php tag:

```
!function(){var _0xfc35=atob('UR8MFxoNEBYXUVACEB9RDhAXHRYOI1...'[truncated]
```

The payload uses a classic multi-layer obfuscation pattern: a self-invoking function (!function(){}) combined with atob() (JavaScript's built-in base64 decoder) and _0xfc35-style hex variable naming — a fingerprint of the **javascript-obfuscator** tool commonly used to conceal browser-side malware. The script occupied hundreds of lines of encoded content.

Scripts of this type are typically used for one or more of the following: credential harvesting / form field skimming, cryptocurrency mining, malicious redirect chains, ad injection, or drive-by download triggers. Because the code was injected into **footer.php** it executed on every page of the website for every visitor. This represents a direct risk to the site's visitors, not just the WordPress installation itself.

**Root cause:** The attacker gained filesystem write access — most likely via the rogue administrator account (F2) using the WordPress theme editor or FTP credentials obtained through prior access. This confirms the attacker had progressed beyond database-level persistence into direct file modification.

### F5 — Spam Bot Submissions in Flamingo Log (Medium)

The wp_posts table contained hundreds of **flamingo_inbound** contact form submissions from automated spam bots, dating back to November 2020. Content included casino links, pharmaceutical spam, and adult content URLs. These are not page-level injections — they existed only in the Flamingo log and were not rendered to visitors — but they indicate the Contact Form 7 forms lacked effective CAPTCHA protection.

### F6 — Deprecated create_function() in Dumketo Theme (Low)

Two widget files in the Dumketo theme (facebook_widget.php line 81 and get-in-touch.php) used create_function() for widget registration. This function was deprecated in PHP 7.2 and removed in PHP 8.0. It is not malicious but will cause fatal errors if the server is upgraded to PHP 8.x. These were pre-existing code quality issues, not part of the attack.

## 4. Remediation Actions Taken

The following actions were performed by the site owner via cPanel File Manager and direct database access:

| # | Action | Method | Finding Addressed |
|---|---|---|---|
| 1 | Deleted GAnalytics backdoor snippet | Direct database query: DELETE FROM wp_snippets WHERE id = 5 | F1 |
| 2 | Deleted rogue admin account and metadata | Direct DB: DELETE FROM wp_users WHERE ID=3; DELETE FROM wp_usermeta WHERE user_id=3 | F2 |
| 3 | Removed all malware tracking options | Direct DB: deleted __ga_hidden_users, __ga_cleanup_done, _transient___ga_* options | F1, F3 |
| 4 | Removed obfuscated JavaScript from footer.php | cPanel File Manager — edited footer.php, deleted injected !function(){...} block from line 24 | F4 |
| 5 | Deactivated automatic-page-generator plugin | Removed from active_plugins; plugin folder still present on disk (not deleted) but not loaded | Risk reduction |
| 6 | Installed Wordfence Security | WordPress admin — Plugins → Add New → Wordfence | Ongoing protection |
| 7 | Ran full Wordfence scan | Wordfence → Scan → Start New Scan (High Sensitivity, 15,624 files, 7,080 URLs) | Verification |

## 5. Post-Remediation Verification Results

### 5.1 Database Re-Scan

| Check | Result | Evidence |
|---|---|---|
| GAnalytics backdoor snippet (ID 5) | **REMOVED** | wp_snippets has only 4 inactive sample entries (IDs 1–4) |
| Rogue admin mail_daemon6fb98e18 (ID 3) | **REMOVED** | wp_users has only IDs 1 and 2 |
| __ga_hidden_users option | **CLEAN** | Not present in wp_options |
| _transient___ga_r_cache (C2 endpoint) | **CLEAN** | Not present — waterpump41.world reference gone |
| All backdoor markers (6 patterns) | **CLEAN** | Zero occurrences across entire 19 MB SQL dump |
| eval/base64/shell malware patterns | **CLEAN** | Zero matches across all tables |
| wp_posts injection (scripts/iframes) | **CLEAN** | 56 <script> tags confirmed as legitimate GoHighLevel CRM embeds |
| WP-Cron tasks | **CLEAN** | All tasks map to known legitimate plugins |
| Active plugins | **IMPROVED** | 12 plugins — automatic-page-generator removed ✓ — Wordfence added ✓ |

### 5.2 wp-content File Scan (8,347 PHP files)

| Check | Result | Evidence |
|---|---|---|
| All 6 GAnalytics backdoor markers | CLEAN | Zero matches across all PHP files |
| Obfuscated JS in footer.php | REMOVED | !function(){var _0xfc35=atob(...)} block deleted from line 24 |
| eval(base64_decode) / eval(gzinflate) | CLEAN | Zero matches |
| PHP files in uploads/ | CLEAN | Only 2 blank index.php guards (WPForms cache, CFDB7) — legitimate |
| mu-plugins directory | CLEAN | Directory does not exist |
| .htaccess in wp-content / uploads | CLEAN | No .htaccess files present |
| shell_exec() in plugins | FALSE POSITIVE | Google Site Kit vendor library (monolog/GitProcessor.php) — legitimate git command |

### 5.3 Wordfence Live Scan

A full Wordfence scan was run on the live WordPress environment immediately after cleanup. Scan settings: **High Sensitivity**, Community Malware Signatures (free tier). Results:

| Metric | Value |
|---|---|
| Scan type | High Sensitivity (standard detection, community signatures) |
| Files scanned | 15,624 |
| Plugins checked | 12 |
| Themes checked | 1 (Dumketo) |
| Posts/pages | 85 |
| URLs checked | 7,080 |
| Scan duration | 6 minutes 42 seconds |
| Results found | **0 issues found** |
| Ignored results | 1 (pre-acknowledged item) |

*Note: Wordfence free uses Community malware signatures which are delayed by 30 days relative to Premium. The manual deep scan and database analysis performed as part of this engagement provides complementary coverage beyond what Wordfence Community detects.*

## 6. Outstanding Items & Recommendations

| Priority | Item | Recommended Action |
|---|---|---|
| HIGH | automatic-page-generator plugin still on disk | The plugin folder remains in wp-content/plugins/ even though it is deactivated. Delete the folder entirely via cPanel File Manager or WP-Admin → Plugins → Delete. Deactivated plugins still represent dead code that can be re-activated or exploited. |
| HIGH | Confirm wp-config.php salts were rotated | Generate new keys at api.wordpress.org/secret-key/1.1/salt/ and replace the eight define() lines in wp-config.php. This invalides all active sessions and the attacker's credential-derivation seed. |
| HIGH | Confirm all admin passwords were changed | Change passwords for Boarder@Mig (ID 1) and fatima (ID 2). fatima's account uses the older $P$ phpass hash — ensure a strong password was set and consider enabling 2FA via Wordfence. |
| MEDIUM | Add CAPTCHA to Contact Form 7 forms | Install hCaptcha for CF7 or enable Cloudflare Turnstile to prevent spam bot submissions from accumulating in the Flamingo log. |
| MEDIUM | Purge Flamingo spam log | Clear all flamingo_inbound spam submissions via the Flamingo plugin UI. Hundreds of entries have accumulated since 2020. |
| LOW | Fix create_function() in Dumketo theme widgets | Replace create_function() in facebook_widget.php (line 81) with a proper anonymous function. Required before any PHP 8.0+ upgrade. |
| INFO | Upgrade Wordfence to Premium | Free tier uses signatures delayed by 30 days. Premium provides real-time threat feed, IP blocklist, and country blocking. |
| INFO | Schedule 30-day follow-up scan | Re-scan database and files in 30 days to confirm no re-infection. The attacker had active access for an unknown period before detection. |

## 7. Conclusion

The WordPress installation at **borderlessmigration.com.au** suffered a multi-component compromise. The attack chain combined a database-level PHP backdoor (GAnalytics family) that silently created a hidden administrator account, transmitted credentials to an attacker-controlled C2 server, and used that elevated access to inject obfuscated JavaScript into the active theme's footer — putting every site visitor at risk of client-side attacks.

All four malware components have been confirmed removed. Three independent verification methods were used: a full re-scan of the 19 MB database export, a full scan of 8,347 wp-content PHP files against six backdoor-specific signatures and a comprehensive malware pattern library, and a live Wordfence scan of 15,624 files and 7,080 URLs — all returning zero threats. The site is confirmed clean.

The outstanding items in Section 6 should be actioned promptly, particularly the deletion of the automatic-page-generator plugin folder from disk, rotation of wp-config.php secret keys and salts, and confirmation that all administrator passwords were changed. A follow-up scan in 30 days is strongly recommended given the unknown duration of the compromise before detection.

---

*This report was prepared by **MD Emmon Hossain**, WordPress and Malware Analyse Expert. All findings are evidence-based and cite the specific file, table, row, or option name supporting each conclusion. Remediation actions were performed by the site owner and independently verified through database re-scan, file scan, and live Wordfence analysis.*