INCIDENT ANALYSIS · JUNE 2026

How One Compromised Reseller Account Let an Attacker Hit Dozens of Websites at Once

An Indonesian gambling-spam campaign called LEMON212 planted doorway pages across multiple unrelated customer sites on our server — not by breaching them individually, but by taking over a single reseller account and walking its customer list.

By **Tremhost Infrastructure Security** · Harare, Zimbabwe · 24 June 2026 · Report version 1.0 · All sources retrieved 24 June 2026

Earlier this month we discovered a coordinated parasite-SEO attack on one of our managed cPanel/WHM servers. Dozens of customer websites — a hospital here, a school there, completely unrelated businesses — had been silently filled with Indonesian online-gambling content. The attacker wasn't targeting any of those businesses individually. They had taken over the account of a reseller who managed all of those sites, and used that single login to walk straight into every site the reseller owned.

We're publishing what we found because the propagation mechanism is underappreciated, the detection lessons are non-obvious, and the mitigations are straightforward once you understand the attack surface.

> *"Operating-system-level isolation did not and could not prevent this spread, because reseller access is a legitimate management channel that operates above the filesystem-isolation layer."*

What is parasite SEO?

Parasite SEO — Google formally calls it *site reputation abuse* — is the practice of planting content on a legitimate, high-authority domain to borrow its ranking signals. Instead of building a credible website from scratch, spammers quietly add pages to reputable sites their targets already trust. Google has a dedicated policy against it and has taken public action to detect and demote it [1].

The variant most commonly observed in the wild right now involves Indonesian online-gambling promotion: terms like *slot gacor* (roughly, "easy-winning slots") and *judi online* (online gambling). Because all forms of gambling are illegal in Indonesia and enforcement has intensified, operators rely on disposable infrastructure and stolen domain authority to keep their offers visible in search [3]. Security firm Sucuri describes this as the single most common SEO spam variant they currently detect [3].

The LEMON212 campaign: what we found on our server

THE DOORWAY PAGES

The injected files were large HTML documents — roughly 150 to 380 KB each — declaring `<html lang="id">` (Indonesian) and carrying a spoofed analytics block imitating a well-known Indonesian news brand. Each page contained dozens of gambling keywords, a campaign identifier (**LEMON212**) embedded in a data-layer field, and a JavaScript loader pointing to external attacker infrastructure. Some doorways replaced

legitimate pages outright (page hijacking); others were added as new files with innocuous-sounding names designed to blend in.

Notably, the attacker did not restrict themselves to `.html` files. Doorways were planted in `.php` and `.txt` files as well — a detail that matters considerably for detection.

SEARCH CONSOLE OWNERSHIP CLAIMS

Alongside the gambling content, the attacker planted Google Search Console verification files (`google<hex>.html`) in web roots. These are used to assert ownership of a domain inside Google Search Console — giving an attacker visibility into, and limited control over, how a domain appears in search. The same verification tokens appeared across multiple, otherwise-unrelated customer accounts: a strong signal of a single actor. Some accounts carried only these tokens, with no gambling content present at all, suggesting the attacker was in the process of claiming properties they hadn't yet filled with payload [2].

> **EARLY WARNING SIGN FOR SITE OWNERS**
>
> Google sends an email notification whenever a new owner is added to a Search Console property. If you receive an unexpected "new owner added" email for a domain you manage, treat it as an active intrusion indicator and check your web root immediately for verification files you didn't place there.

Root cause: one reseller login, many victims

The defining feature of this incident is that it was not a set of independent site break-ins. CONFIRMED A full server review established that the overwhelming majority of affected accounts shared a single owner: a reseller account. Access logs showed authenticated WHM sessions originating from

two IP addresses in Indonesian IP space, performing account-enumeration and account-transfer operations — in other words, an attacker operating the reseller's control panel as if they were the legitimate reseller.

This explains why the victims were unrelated organisations across different sectors, hit in waves. The attacker wasn't looking for a vulnerability in each site; they were simply walking a reseller's customer list.

The server had filesystem-level account isolation active — a control that prevents one account's code from reading another's files. **That isolation did not prevent this spread, and it couldn't have.** Reseller WHM access is a legitimate authenticated management channel that operates above the filesystem-isolation layer. Per-account password rotation would equally have been irrelevant: the attacker entered through the reseller privilege tier, not individual customer logins. Current security literature consistently notes that compromise of a WHM or reseller account is effectively a compromise of every site that account serves [4][5][6].

HOW THE RESELLER ACCOUNT WAS LIKELY COMPROMISED

ASSESSED The most probable initial-access route was plaintext credentials stored in the reseller's own web root. We found application configuration files (`.env` files) containing reused database passwords and live mail passwords, stored world- or group-readable. A subsequent server-wide check confirmed this misconfiguration was not isolated to the compromised account. We classify this as *assessed* — the exposure is confirmed, but we did not recover a specific log entry tying it directly to the credential theft. We found no evidence of any published cPanel/WHM vulnerability being exploited; the available evidence points to a credential-based entry [4].

How we detected it: lessons on signatures

Getting detection right required two corrections to our initial approach.

Heuristic searches produced too many false positives. Generic terms — a data-layer field name, the word "slot" — matched legitimate application code, mail archives, and templates, badly overstating scope. What worked reliably was a *literal, fixed-string* search for the unique JavaScript loader token embedded in every doorway page. That search returned zero false positives across the entire server. A secondary structural signature — `<html lang="id">` — was also useful, since a business or healthcare site in our market is essentially never authored in Indonesian. But the loader token was the reliable primary signature.

Restricting searches to `.html` files caused misses. Because doorways were planted in `.php` and `.txt` files, a whole-filesystem, all-file-type sweep was required to bound scope with confidence.

Indicators of compromise

We publish these so other operators can detect and act on this campaign more quickly. All attacker-controlled artefacts are included; all customer identities, internal hostnames and credentials are omitted.

CATEGORY	INDICATOR
Payload loader (reliable)	A fixed JavaScript loader token string in a `<script src>` on every doorway. Detect via fixed-string search across all file types; no false positives observed.
Doorway page markers	`<html lang="id">`; spoofed Indonesian-news analytics block; campaign field value **LEMON212**; heavy "slot"/"gacor" keyword density; file size ~150–380 KB.

CATEGORY	INDICATOR
File placement	Doorways in `.html`, `.php`, and `.txt`. Mix of new files and overwritten legitimate pages, named to mimic real content.
Ownership-claim artefacts	Google Search Console verification files (`google<hex>.html`). Identical tokens recurring across unrelated accounts indicate a common actor.
External infrastructure	A Cloudflare Pages deployment used as attacker infrastructure (subdomain containing the campaign name on `pages.dev`).
Reseller-abuse pattern (logs)	Authenticated WHM sessions performing `listaccts` (account enumeration) and `xfercpanel` (account transfer) from attacker IPs in Indonesian IP space.
Initial-access exposure	World/group-readable `.env` files in web roots containing plaintext database and mail credentials.

Containment and recovery

We followed a standard sequence: sweep the full server with reliable signatures to bound true scope; suspend every account confirmed to be serving payload; remove and preserve the attacker's Search Console verification files; block the attacker source addresses; rotate affected credentials.

Two points are worth calling out explicitly for other operators.

The reseller credential must be rotated even while the account is suspended. On cPanel, a suspended account's password cannot be changed directly — doing so unsuspends it. The safe procedure is a tightly-scoped *unsuspend → change-password → re-suspend* sequence, executed as a

single chained operation with the attacker's source addresses already blocked so the account is never meaningfully exposed during the rotation window.

Rotating the credential is necessary but not sufficient. If the underlying cause — plaintext or reused secrets in a web-accessible location — is not corrected, a fresh credential can leak by the same path.

For the affected sites themselves, recovery was often tractable without a full restore. Where the attacker replaced only a site's front controller (for example, a WordPress `index.php`), the genuine content remained intact in the database and the site could be restored by reinstating the standard front controller and removing doorway files. Static sites required doorway removal plus manual rebuilding of overwritten pages. Where clean backups were available, restoring to a known-good snapshot was preferable.

> **SEARCH CONSOLE: DELETING THE FILE IS NOT ENOUGH**
>
> Removing the attacker's verification file un-verifies that token over time, but does not automatically evict an already-verified owner. Affected domain owners must explicitly remove the unrecognised owner inside Search Console — this is a distinct remediation step that the hosting provider cannot perform on the customer's behalf.

What hosting providers should do

01. **Mandate multi-factor authentication on all reseller and WHM accounts.**
    This single control most directly defeats the mechanism described here. A stolen credential without a second factor is useless. This is the top recommendation in current guidance and removes one of the most common compromise paths at minimal operational cost [4].

02. **Maintain and regularly test backups for every account.**

Restorable, verified backups let a compromised site return to a known-clean state rather than be rebuilt by hand. Test a full single-account restore end-to-end, not merely confirm that files exist [4].

---

03. **Detect and prohibit plaintext credentials in web roots.**

A scheduled scan for world/group-readable `.env` and configuration files surfaces exactly the exposure that most plausibly seeded this incident.

---

04. **Alert on reseller account-enumeration and account-transfer operations.**

`listaccts` and `xfercpanel` commands from unexpected source addresses in WHM logs are high-value telemetry. Consider source-restricting management logins and reviewing reseller privileges against actual need.

---

05. **Use per-account PHP isolation as defense-in-depth.**

Per-domain PHP-FPM and `open_basedir`, alongside filesystem-level account isolation, raise the cost of lateral movement at the application layer [4]. Note that none of these address the reseller-privilege tier — that requires authentication controls, specifically MFA.

---

06. **Include Search Console hijack in your incident response runbook.**

Instruct affected domain owners to audit and remove unrecognised Search Console owners as a distinct step, not assume that deleting the verification file is sufficient [2].

Conclusion

Parasite-SEO gambling spam is industrialised and financially motivated. This incident illustrates how efficient a hosting reseller account can be as a delivery vehicle: by compromising one credential, an attacker reached and

defaced many unrelated customer sites at once, entirely unimpeded by per-account filesystem isolation that was working exactly as designed.

The practical takeaways: use specific, exhaustive signatures that run across all file types; treat the reseller and WHM privilege tier as a distinct attack surface requiring its own controls (foremost, mandatory MFA); and maintain tested backups so that recovery is a restore, not a rebuild.

We publish the indicators above so that other operators can detect and remediate this campaign more quickly. If you identify the LEMON212 campaign on your infrastructure and have artefacts to share, we welcome coordinated disclosure.

References

1. Google, "Defending Search users from 'Parasite SEO' spam," *The Keyword*, 2025. blog.google
2. Jonias Fortuna (Indocenter & Radnext), "Anti-Gacor: Prevention and Protection Against Online Gambling Slot Injection on Websites," *potato.id*, 2026. potato.id
3. Sucuri, "Slot Gacor: The Rise of Online Casino Spam," *Sucuri Blog*, 2025. blog.sucuri.net
4. HostMyCode, "cPanel Security Checklist Tutorial (2026): Audit WHM, Accounts, and Services," 2026. hostmycode.com
5. CyberSecureFox Editorial Team, "cPanel Fixes High-Risk WHM and WP Squared Flaws," 2026. cybersecurefox.com
6. The Hacker News, "Critical cPanel Authentication Vulnerability Identified — Update Your Server Immediately," 2026. thehackernews.com

All URLs retrieved and verified 24 June 2026.