SECAUDIT 050 - CMS architecture assessment: direct-access hardening + content-model integrity

SECURITY AUDIT / PUBLIC REMEDIATION RECORD

Status: **one security finding closed in code (live verification pending); one data-integrity finding verified; architecture findings dispositioned, remainder deferred with owners.**

Why this audit happened

SECAUDIT 047 and 048 reviewed the code line by line and against a live target and found it strong. 050 is deliberately different: a structural review of the whole CMS against the patterns mature publishing, media-library, database, recovery, security and federation systems follow - the questions a code review does not ask. It produced one closable security finding, one data-integrity verification, and a foundational finding an earlier review should have caught. This report covers current development code; per-installation and fleet verification of the shipped fix is recorded as outstanding, not complete.

Finding 1 - Core includes were reachable directly; the deny-list had drifted (FIXED in 0.7.544, live-verify pending)

Severity: MEDIUM (defense-in-depth / trust-boundary). Not remote code execution and not, on its own, a data leak - `core/db.php` does not echo - but a set of sensitive server-side includes were reachable as their own URLs on any host that does not honour Apache rules.

Mechanism. Only `core/smackback.php` carried a PHP execution guard (a `defined('SNAPSMACK')`-style check). Every other core include depended on a single Apache `<FilesMatch>` / `.htaccess` deny-by-name list. On nginx, or any host that ignores `.htaccess`, that list evaporates and the includes are directly requestable. The reachable set included `core/updater.php`, `core/release-pubkey.php` (the release-signing public key), `core/skin-registry.php`, `core/manifest-inventory.php` (the shared-asset registry) and the real authentication include `core/auth-smack.php`.

Two things made it worse than a clean deny-list would have been:

- **The list had drifted from the code.** It still named `login` and `auth` - entries

that no longer correspond to shipping files - while the authentication file that actually matters, `auth-smack.php`, was not listed at all.

- **Routing is default-open.** The filesystem `!-f` passthrough means every root

`.php` is an endpoint unless something denies it, so the deny-list was the only thing standing between a request and these includes.

Fix (0.7.544). Each of the five includes above now carries a server-agnostic PHP direct-access guard that returns 403 and exits when the file is requested outside the application, independent of web server. The Apache `<FilesMatch>` list was reconciled to the filenames that actually ship - the dead `login`/`auth` entries removed, the real `auth-smack.php` added. `core/db.php` is deployment-local (its contents differ per install), so its guard is applied server-side rather than shipped.

Status. The code is released in 0.7.544. It reaches installations through the normal tag -> updater -> System Maintenance *Repair* path; confirming the guards respond 403 on each live install is the outstanding verification step and is not marked done.

Finding 2 - Orphaned-engagement integrity check (VERIFIED: zero true orphans; a diagnostic error corrected)

Severity: data integrity, no security exposure. A long-open check for community records - comments, likes, reactions - with no valid parent.

Method. Read-only, offline, against the migrated `foreverphotograph.ing` database dump (the highest-risk site: the most community data, and migrated). The definition used matches how the code actually reads engagement: a record is orphaned only if its key exists in **neither** `snap_images.id` **nor** `snap_posts.id`.

Result. Zero true orphans - comments 3819 -> 0, likes 4 -> 0, reactions 0; the legacy `snap_comments` table empty. A consistency guard confirmed 9,975 images against 9,975 posts.

The corrected error. An earlier diagnostic used a **posts-only** definition - requiring every comment to key to `snap_posts.id` - which is wrong for this system: comments and likes key to the **image** identifier by design, and the display reads them that way. That assumption mis-classified ~1,904 valid community comments as orphans and they were briefly removed during analysis, then restored from the continuity dump before the corrected, display-aware check confirmed there was no orphan problem. Recorded lesson: an integrity check must model how a table is actually read, not how one assumes it is keyed.

Finding 3 - No enforced definition of a publishable content unit (architecture; remediation in progress)

Severity: architectural / correctness. The foundational finding, and one an earlier audit missed while reporting the code clean.

Mechanism. The CMS had no single, enforced definition of "a published item." Legacy solo publishing (SMACKONEOUT) stored a photograph as a bare `snap_images` row and treated that row as the post; the newer group and longform paths used a separate `snap_posts` record. With two representations, publication state, dates, ownership, engagement identifiers, collections and federation identity had no single home and could take on competing meanings. Clean parameterisation, escaping and authorization do not detect this - it is a data-model question, which is why a code review passed it and an architecture review is where it surfaces.

Remediation in progress. Releases 0.7.534D-0.7.536D added post-model inventory and transactional conversion tooling. The first conversion diagnostic then over-corrected in the opposite direction - it treated valid images sitting in a draft longform bucket as old-style posts. Current source reports bucketed and unassigned longform media separately and prohibits photo-to-post conversion outside SMACKONEOUT. This work is not represented as finished.

Disposition summary

The review dispositioned eight findings. Two are closed and disclosed here (Finding 1, fixed in code; Finding 2, verified). Finding 3's remediation is in progress. The remaining findings - including further hardening of administrator authorization and the consolidation of URL-to-handler routing - are deferred with named owners and target windows. They are deliberately **not** described in operational detail in this public record: publishing the location and mechanism of an unremediated weakness is itself a risk. Each will be disclosed here, concretely, once it is closed - the same way Finding 1 is disclosed now.

Process correction - the review method, not just the finding

050 exposed a process failure that mattered more than any single finding: asked to compare the architecture against similar platforms, the earlier review answered from internal knowledge, found nothing, and reported the system sound - while the CMS had no post object at all, which every mainstream CMS has. Disclosing that miss is not the same as preventing the next one, so the review standard itself was corrected, not just this finding.

The Minimum Compliance Standard (SMCS) is now v1.1:

- **A Method section (M1-M3).** A comparative review must produce a *cited matrix* - named

comparators (WordPress, Ghost, Drupal, Koken, Pixelfed, ...) against our architecture on each dimension, with sources - not a verdict from memory. Every deviation goes in a register labelled deliberate / accepted-debt / defect, and the review cannot be marked complete, or report "code is clean," until that matrix and register exist and every deviation is dispositioned. A review that consulted no external sources must say so - which reads as incomplete, not clean.

- **A Part D (D1-D4): data-model & content invariants.** One enforced publishable-unit

model with every write path enumerated and converging on it; one documented keying scheme per relationship; integrity checks that model how a table is actually read; and deviations logged rather than discovered later.

Both mistakes from this audit - no external comparison, and an undefined content unit - are now gated checklist items that a future review must clear before it can pass.

Scope and limits

Findings are stated against current development code. Finding 1's per-installation and fleet-wide response-header/guard verification is outstanding. Finding 2 was verified against one site's data (the highest-risk one); a spot check across additional site dumps is recommended before claiming fleet-clean. The deferred items are tracked in the internal charter and finding records and are not reproduced here.