SECAUDIT 034 - Executable skin manifests and redundant credentials export: closure verification

Field	Value
Audit ID	2026-07-24-034
Date	2026-07-24
Severity	CRITICAL (historical) - skin metadata could execute arbitrary PHP; MEDIUM (defense-in-depth) - redundant authenticated exports exposed credential material
Component	Skin manifests, skin package installation, Disaster Recovery, SUYB export
Status	REMEDIATED - RCE path closed in 0.7.440; transition bridge retired in 0.7.444; credential-only exports fully removed for the next release
Reporter	Sean (flagged both unnecessary trust boundaries) + Claude and Codex (traced, remediated, and verified)
Related	006 (release integrity), 017 (SMACKBACK), 025/026 (skin attack surface), 030 (original manifest RCE finding), 031 (remove rather than retain dangerous server-side capability)
Disclosure	No exploitation is known. This report records early design decisions, the questions that exposed their risk, and removal of the unnecessary execution and export paths.

1. Summary

Two early design decisions gave the CMS more power than the feature required.

First, skin metadata lived in `manifest.php`. The file described names, versions, controls, fonts, and required engines, but the CMS obtained that data with PHP `include`. Asking "what happens if a submitted skin puts code in its manifest?" exposed the real trust boundary: the answer was not "it changes some settings"; the answer was arbitrary PHP execution with the web process's privileges. That was an unnecessary Remote Code Execution surface.

Second, Disaster Recovery offered a separate USER CREDENTIALS download. It performed `SELECT * FROM snap_users` and returned the table as SQL. The full Recovery Kit beside it already included the same table, so the smaller export added no recovery capability. It did add another route capable of producing a portable file containing password hashes, recovery data, and potentially 2FA secrets.

The visible Disaster Recovery route was removed in 0.7.440. This closure review then found the same credential-only mode still reachable through the authenticated SUYB export endpoint (`suyb-export.php?type=keys`) and retained inside the shared export engine. That residual route is now removed too. This is why closure reviews exist: removing one button and one handler is not proof that the underlying capability has disappeared everywhere.

Skin metadata is now inert, validated JSON; official skin ZIPs must be signed; ZIP paths are checked before extraction; and the final `manifest.php` transition adapter stopped shipping in 0.7.444 after every known installation reached 0.7.443. Recovery Kit export remains. Credential-only export does not. No exploitation is known.

2. Finding A - executable skin metadata (CRITICAL, closed)

2.1 The early design decision

Skins originally returned their metadata from PHP:

$manifest = include "skins/{$active_skin}/manifest.php"; ```

That was convenient because manifests could compute arrays and consult the shared font and engine inventory. It also meant metadata was not data. It was code. Any statement before the returned array executed on public and administrative requests that loaded the manifest.

The feature required a structured description of a skin. It did not require code execution. The executable format therefore crossed a trust boundary for no necessary product benefit.

2.2 Delivery and impact

A malicious or compromised third-party skin package could place arbitrary PHP in `manifest.php`. Once installed, a normal page request could execute it. Potential impact included:

- reading or changing the database;
- exfiltrating settings, authentication material, or private content;
- writing files and establishing persistence;
- changing administrative controls or hiding evidence;

- acting with every filesystem and database privilege available to PHP.

SMACKBACK could detect later changes to a known-good file, but it could not make malicious-by-design code safe. A malicious manifest matching its own baseline would still execute.

2.3 Remediation

Release 0.7.440 replaced all official skin manifests with `manifest.json` and a single core-owned loader:

- JSON input is size-limited and parsed with exceptions enabled.
- The loader accepts a fixed top-level schema and known control types.
- Unknown or malformed structures fail closed or normalize to safe defaults.
- Manifest values are never passed to `include`, `require`, `eval`, or a callback.
- Gallery and fresh-install skin packages require Ed25519 signatures.
- Missing Sodium, signature, or release public key fails closed.
- ZIP entries are rejected for traversal, absolute paths, drive-letter paths, NUL bytes, and `..` segments before extraction.

During the fleet transition, signed packages contained a tiny generated `manifest.php` adapter that returned decoded `manifest.json` for older cores. It was deliberately temporary. After every known installation reached 0.7.443, release 0.7.444 removed adapter generation and changed all 23 official skin footers to call the JSON loader directly.

3. Finding B - redundant user-credentials exports (MEDIUM, closed)

3.1 What existed

The Disaster Recovery screen presented a USER CREDENTIALS card backed by an authenticated `type=keys` export handler. The handler read the `snap_users` table definition, ran `SELECT * FROM snap_users`, and serialized every row into a downloadable SQL file.

Depending on schema version and enabled features, those rows could contain password hashes, recovery-code hashes, 2FA state, TOTP secrets, and recovery material. Hashes are not plaintext passwords, but they are still security material and should not receive an extra export route without a distinct need.

The first remediation removed the Disaster Recovery card and its POST handler. The closure review found a second route: the authenticated SUYB endpoint still accepted `type=keys`, and the shared export engine still implemented a `snap_users`-only dump.

3.2 Why the routes were unnecessary

The full Recovery Kit and full authenticated SUYB database export already contain `snap_users`. Removing the smaller exports does not reduce disaster-recovery capability.

Each separate route increased attack surface and operational exposure:

- another server handler capable of releasing credential material;
- another way to create a sensitive file in browser or tool storage;
- another response path to audit for authentication, authorization, request forgery, caching, and content-disposition mistakes;
- a misleading impression that credential-only SQL was the preferred recovery artifact.

These were not unauthenticated password dumps. They were authenticated capabilities. Their severity comes from the sensitivity of the material and the complete absence of a unique recovery purpose.

3.3 Remediation

The complete remediation removes every credential-only path:

- the USER CREDENTIALS card and download button;
- the `type=keys` branch in `smack-disaster.php`;
- `keys` from the SUYB endpoint's accepted types and public documentation;
- the `keys` branch from `SnapSmackExport::emitSqlDump()`.

Removing only the visible button would have left a direct request route. Removing only that route would have left the same capability through SUYB. The shared export engine now supports only full and schema SQL dumps; Recovery Kit export/import remains available. Credential recovery belongs in the protected recovery workflow and encrypted Break-Glass Card, not in a convenience dump of the user table.

4. Verification

4.1 Manifest boundary

- All 23 official skin directories contain `manifest.json`; none contains a source `manifest.php`.
- Runtime consumers use `load_skin_manifest()` or `snapsmack_load_manifest()` and do not execute skin metadata.
- All 23 official `skin-footer.php` files lint clean after removal of the transition fallback.
- Local and Smack Central packagers contain no adapter-generation code.

- A GALLERIA package built from 0.7.444 contains `manifest.json` and no `manifest.php`.
- Registry installation requires `manifest.json`, mandatory signature verification, and safe ZIP paths.

4.2 Credentials-export boundary

- `smack-disaster.php` contains no `type=keys` handler or USER CREDENTIALS card.
- `suyb-export.php` rejects `type=keys` as an invalid export type.
- `core/export-engine.php` contains no credential-only table-selection branch.
- Repository search finds no live `type=keys` credential export route.
- Recovery Kit and full authenticated database export remain available.

5. Residual trust and policy

Skin metadata no longer executes, but a skin still contains PHP templates and JavaScript. Installing a skin therefore remains a code-deployment decision. Mandatory signatures establish who built the package and protect its bytes in transit; they do not turn executable templates into harmless content.

Official Gallery and fresh-install packages remain signature-gated. Owner-authoring surfaces such as OH SNAP retain their deliberately scoped code-deployment power and must continue to require narrowly issued credentials. Future third-party skin intake should treat templates as code, manifests as data, and every new server-side export as guilty until it demonstrates a recovery purpose not already served by the Recovery Kit.

6. Closure

The manifest issue was an early design decision flagged during an ordinary security question, not evidence of malicious intent or known exploitation. The important result is that the trust boundary was recognized, documented, fixed, deployed across the fleet, and stripped of its temporary rollback bridge.

The credentials export followed the same rule: if a powerful path is redundant, remove it rather than polish it. The closure review found and removed the second route instead of claiming the first removal had finished the job.

SnapSmack retains the recovery capability and loses the unnecessary attack surface.

Disposition: CLOSED.