SNAPSMACK SECURITY AUDIT

Session Changes: SYBU Recovery + Unzucker Caption Fix — #024

Date: 2026-06-12
Components: SYBU (Smack Your Batch Up) 0.7.9k · Unzucker importer · core 0.7.254
Scope: `tools/sybu/recovery.py` (new), `tools/sybu/main.py`, `tools/sybu/poster.py`, `tools/unzucker/poster.py`, `tools/unzucker/ig_parser.py`. Server-side sinks (`core/snap-tags.php`, `skins/the-grid/layout.php`, `core/unzucker-api.php`) reviewed only where this session's changes move or widen a trust boundary.
Auditor: Claude (Opus 4.8), Cowork session — for Sean McCormick
Note: Claude Commander changes are excluded here — that directory is gitignored from snapsmack and is covered separately in `_continuity/claude-commander-audit-2026-06-12.md`.

EXECUTIVE SUMMARY

This session changed two desktop client tools and touched **no server endpoint, auth path, or new parameter**. SYBU gained a local crash-recovery cache, per-row batch selection, an in-progress job cancel, and a fix to a broken pre-POST session check. Unzucker's importer got two caption fixes (stop duplicating the caption into the post title; fall back to the media-level caption for single-image posts).

No critical or high findings. One finding carries real weight: the Unzucker `ig_parser` caption fallback (**UZ-11**) widens the set of posts whose caption text reaches `snap_render_caption_html()` — a **pre-existing stored-XSS sink** that passes whitelisted <a> tags through with their attributes intact via `strip_tags`. This session did not create that sink, but it increases exposure to it; the correct fix is at the sink and benefits the entire site. Every other change is local-only and rates Low or Informational. The SYBU session-check removal is explicitly **not** a security regression.

EOF MARKER / TRUNCATION CHECK

File	EOF Marker	Status
`tools/sybu/recovery.py` *(new)*	`# ===== SNAPSMACK EOF =====`	✓ OK
`tools/sybu/main.py`	`# ===== SNAPSMACK EOF =====`	✓ OK
`tools/sybu/poster.py`	`# ===== SNAPSMACK EOF =====`	✓ OK
`tools/unzucker/poster.py`	`# ===== SNAPSMACK EOF =====`	✓ OK
`tools/unzucker/ig_parser.py`	`# ===== SNAPSMACK EOF =====`	✓ OK

All 5 changed source files carry valid EOF markers (verified via the Read tool, not the Linux mount, which served truncated copies this session). No truncations.

FINDINGS

UZ-11 — Caption fallback widens exposure to a stored-XSS sink

MEDIUM

Change: `tools/unzucker/ig_parser.py — parse(), media[].title fallback` | **Sink:** `core/snap-tags.php — snap_render_caption_html()` ← `skins/the-grid/layout.php L247`

Data flow: `media[].title` → `ParsedPost.caption` → Unzucker POST → `snap_posts.description` (stored via prepared statement — no SQLi) → rendered by The Grid through `snap_render_caption_html()`. That function calls `strip_tags($text, '<p><br><strong><em><u><a><ul><ol><li><blockquote>')`. `strip_tags` removes disallowed tags but keeps whitelisted tags **with their attributes intact** — so `<a href="javascript:…">` or `<a onmouseover="…">` survives into the page. This is a classic `strip_tags` attribute-passthrough stored-XSS vector.

This sink is **pre-existing** — carousel posts always carried captions through it. This session's fix adds single-image posts (which previously imported blank) to the same flow, so it **widens the population** of posts whose caption text reaches the sink. Likelihood is low in normal use (Instagram captions are plain text), but non-zero: a hand-edited or third-party "poisoned" export (see UZ-03, audit #023) could embed markup in `media[].title`. Impact is stored XSS against any viewer of that blog.

Recommendation: Fix at the sink, where it protects every caller: in `snap_render_caption_html()`, `htmlspecialchars()` the text first and then run the hashtag linkifier (which already emits safe escaped anchors), or drop `<a>` from the `strip_tags` whitelist, or run a real sanitizer that strips attributes/event handlers from allowed tags. Pair with the UZ-03 export-root constraint to reduce untrusted-export intake.

UZ-12 — poster.py title='' reduces stored caption duplication

IMPROVEMENT

Change: `tools/unzucker/poster.py — create_post(title='')`

The importer now sends an empty post title instead of the caption, eliminating the double-write that previously placed the full caption in both `snap_posts.title` and `snap_posts.description`. The title render path (`layout.php` L244) was already `htmlspecialchars`-escaped and so was safe, but this change still **reduces** the amount of user-controlled caption text stored and rendered — one fewer field carrying it. The unchanged `prepare_image(title=post.body)` on L361 writes EXIF metadata into the image file, not a web sink, and is out of scope of the XSS path. Net positive; no new surface.

SY-01 — Removed pre-POST session check is not a security regression

LOW

Change: `tools/sybu/main.py — _ensure_connected()`

The deleted `self._client.is_session_alive()` call was dead code (it raised `AttributeError` after the 0.7.9e API-key migration removed the method — the cause of the 0.7.9j "Checking session…" hang). Under API-key auth there is **no server session to validate**: the `X-Snap-Key` header is transmitted and verified by the server on every request regardless of any client-side state. Removing the

broken client check restores function without weakening server-side authentication. Documented for completeness; no action.

SY-02 — Recovery cache written unencrypted and trusted on load (local-only) LOW

Change: `tools/sybu/recovery.py — RecoveryStore`

The new recovery file `recovery/sybu_recovery_<jobid>.json` stores post metadata only — titles, tags, categories, colors — and **no credentials**. Writes are atomic (temp + fsync + `os.replace`); loads fall back safely on a missing/corrupt file. There is no path-traversal exposure: the jobid is a SHA-1 hash, the filename pattern is fixed, and the stored `entry.file` is data only (never used as a filesystem path). No network, no `eval`. The residual risk is purely local: an attacker who already has write access to the recovery directory could pre-seed or alter enrichment that is later posted to the site — a data-integrity concern on an already-compromised host, not code execution.

Recommendation: Accept (local-only, non-credential). Optional hardening: validate field types and lengths when restoring from the recovery file, treating it as untrusted input even though it is local.

SY-03 — Recovery directory may be unwritable under protected install paths INFORMATIONAL

Change: `tools/sybu/recovery.py — _recovery_dir()`

The recovery directory is created next to the executable. If SYBU is installed under a protected location (e.g. `Program Files`), directory or file creation can fail; the code swallows the `OSError` and silently skips the save. This is a robustness/usability gap (the recovery feature quietly no-ops), not a security issue. Recommendation: consider `%LOCALAPPDATA%` for the recovery store and surface a one-time warning if writes fail.

SY-04 — Cancel and batch-selection changes carry no security impact INFORMATIONAL

Change: `tools/sybu/main.py (selection, cancel) · tools/sybu/poster.py (cancel_event)`

Per-row selection and the `threading.Event` cancel alter only which and how many local queue items are processed. Each server post remains atomic; cancelling mid-batch leaves already-posted items posted and stops cleanly between images, with no partial-record corruption server-side. No new input, endpoint, or authentication path is introduced.

FINDINGS SUMMARY

ID	Severity	Title	Status
UZ-11	MEDIUM	Caption fallback widens exposure to stored-XSS sink (strip_tags attribute passthrough)	OPEN (sink pre-existing)
UZ-12	IMPROVEMENT	title='' reduces stored caption duplication	RESOLVED

SY-01	LOW	Removed session check — not a security regression	ACCEPTED
SY-02	LOW	Recovery cache unencrypted, trusted on load (local-only)	OPEN (by design)
SY-03	INFORMATIONAL	Recovery dir may be unwritable under protected install paths	OPEN
SY-04	INFORMATIONAL	Cancel / selection changes — no security impact	INFORMATIONAL

> **Verdict.** The session's changes introduce no new web-facing attack surface and no new auth path; SYBU's are local-only and Low/Informational, and the session-check removal is a fix, not a regression. The one item to act on is **UZ-11**: the Unzucker caption fallback is correct and should ship, but it increases exposure to a pre-existing stored-XSS sink in `snap_render_caption_html()`. Prioritize hardening that function (escape-then-linkify) — it closes the vector for every caption on the site, Unzucker-imported or not. Ship the fixes; track UZ-11 as the next hardening task, alongside UZ-03 from audit #023.
