SNAPSMACK_EOF_HEADER Last non-empty line of this file MUST be the canonical EOF marker for this file type: an HTML comment containing five equals, space, the literal string 'SNAPSMACK EOF', space, five equals. Missing or different = truncated/corrupted. Restore before saving.

SECAUDIT 004 - closure record

SECURITY AUDIT / PUBLIC REMEDIATION RECORD

Field	Value
Closes	`2026-04-26-004-suyb-security-audit` - Smack Up Your Backup, first review, v0.7.9h
Date	2026-08-07
Verified against	SUYB at 0.7.19+, working tree at `d0e2d64b`
Result	**4 of 5 findings closed. 1 accepted by documented decision.** No finding remains open through oversight.
Reporter	Claude, re-verifying each April finding against present code rather than against later audit prose

Why this document exists

The April 2026 review was SUYB's first, and every finding in it was recorded as `OPEN` because at the time they all were. Those items were subsequently fixed - mostly by SECAUDIT 037 - but **the original report was never updated and never published**, so the only public record of SUYB's first audit was no record at all.

It could not simply be posted as-is. Its own text lists a HIGH finding, a Google service-account private key, as open, along with an instruction to rotate the key and rewrite git history. Publishing that in August, when it had long since been handled, would frighten readers about a live exposure that does not exist.

So this is the closure: every original finding restated, with its disposition **verified against the code as it stands today**, not inherited from a later audit's summary. Where the answer is "we decided not to", it says so.

Finding 1 - Google service-account private key in the repo directory - CLOSED

Original severity: HIGH. Original status: OPEN.

The April review found a Drive service-account key (`suyb-drive-key-*.json`) sitting in the tool directory, with that directory's `.gitignore` covering only `*.pyc` and `__pycache__/`. It warned - correctly - that a key committed even once must be treated as compromised, because a repository that has ever been pushed makes its history public in effect.

Verified today, three ways:

```
git log --all --full-history -- "tools/smack-up-your-backup/*drive-key*.json"
-> no commits. The key was NEVER committed, on any branch, at any point.

ls tools/smack-up-your-backup/*drive-key*.json
-> not present in the tool directory.

git check-ignore -v tools/smack-up-your-backup/suyb-drive-key-5e7a5909f75e.json
-> tools/smack-up-your-backup/.gitignore:6 *-drive-key-*.json
```

A repo-wide search for private-key material (`BEGIN ... PRIVATE KEY`) across every tracked file also returns nothing.

The important nuance: the compromise scenario never applied. The April report called for immediate key rotation and a `git filter-repo` history rewrite *if* the file had ever been committed. It had not been. The exposure was a key sitting in a working directory that was one careless `git add -A` away from being published - a real risk, and not the same thing as a leaked key. No rotation was required and none is required now.

The recommended ignore pattern was adopted verbatim.

Finding 2 - FTP and admin passwords stored as base64 - CLOSED

Original severity: MEDIUM. Original status: OPEN.

Profile files stored the FTP password and the SnapSmack admin password as base64, which the module docstring itself admitted was obfuscation rather than encryption.

Closed by SECAUDIT 037 (SUYB 0.7.19), which introduced `secret_vault` - a passphrase-derived key (scrypt) sealing secrets with Fernet. Verified in `profile_manager.py`: secrets now route through `secret_vault.encrypt()` / `decrypt()`, and the vault **fails closed** - a locked vault refuses to re-save rather than silently writing the weaker form.

The original `_obfuscate()` remains as the fallback for installs that have not enabled the vault. That is deliberate: it keeps existing profiles readable rather than stranding them, and it is strictly what the tool did before. The April recommendation suggested Windows DPAPI; the vault was chosen instead because it is cross-platform and portable, which matters for a tool whose whole design point is that it can be carried between machines.

Finding 3 - FTPS certificate verification off by default - ACCEPTED, NOT CLOSED

Original severity: MEDIUM. Original status: OPEN. Current status: open by decision.

Still `False` by default (`profile_manager.py` new-profile template; `transport.py` passes it through). This is the one item the April review raised that has **not** been fixed, and it is not an oversight.

SECAUDIT 037 §7.1 recorded the reasoning on 2026-08-05: SnapSmack's audience runs on budget shared hosts that routinely ship self-signed or expired certificates. Defaulting to full CA validation would break the majority of real profiles on first connect, and a binary verify/don't-verify toggle trades one bad outcome for another.

The forward path is **not** mandatory CA validation. It is trust-on-first-use certificate-fingerprint pinning, mirroring the SFTP host-key pinning that 037 already shipped (verified today: `sftp_client.py` maintains a portable `suyb_known_hosts` pin file). That detects a machine-in-the-middle without ever requiring a CA-signed certificate - which is the protection the cheap-host reality actually needs.

Recorded here as an open, deliberate, dated decision rather than quietly dropped.

Finding 4 - unescaped `name_filter` in the Drive API query - CLOSED

Original severity: LOW.

Verified in `cloud_client.py`: `DriveClient.list_files()` now escapes the value before interpolation (`safe_filter = name_filter.replace("'", "\\'")`).

The second `list_files()` in the same file belongs to a different provider and filters client-side with a plain Python containment test - no query language, so nothing to inject.

Finding 5 - unconditional debug log next to the exe - CLOSED

Original severity: LOW.

Verified: the inner `_dbg()` function and the `suyb-debug.log` write are gone from `cloud_client.py` entirely. The log had recorded folder ids, query strings and file counts on every user's machine with no rotation.

The three INFO items

All three were marked PASS in April and remain unchanged: OAuth tokens stored beside the credentials file under the narrow `drive.file` scope (standard for desktop OAuth apps), broad MIME acceptance on the SQL dump endpoint, and in-memory credential retention for session re-login.

Summary

#	Finding	April	Today
1	Service-account key in repo directory	HIGH / open	**Closed** - never committed, removed, ignored
2	Passwords stored as base64	MEDIUM / open	**Closed** - scrypt + Fernet vault (037)
3	FTPS cert verification off by default	MEDIUM / open	**Accepted** - documented 037 §7.1, TOFU pinning is the path
4	Unescaped Drive query filter	LOW / open	**Closed**
5	Unconditional debug log	LOW / open	**Closed**

Nothing from SUYB's first review remains open by accident.