SECAUDIT 045 - Pixelix live interoperability and release lifecycle

SECURITY AUDIT / PUBLIC REMEDIATION RECORD

Field	Value
Date	2026-08-13
Scope	Deployed `v0.7.520D` Pixelix adapter, OAuth login with 2FA, media staging, ALT update, status creation, optional Pixelfed compatibility reads, Apache routing, updater/SMACKBACK interaction, and release-tag lifecycle
Baseline	Deployed tag commit `48099d45ee9ecede0476d0760d5d5173b8037a75`, live access/error logs supplied from ALLINTHEWRIST, and the subsequent `dev` remediation
Status	**CLOSED. The live posting contract passed and all five follow-up findings are remediated. The historical 520D tag movement cannot be undone; 520D is retired and the release pipeline now makes future identifiers immutable.**
Positive controls	GRAMOFSMACK-only mode gate; owner Offline Posting gate; password and 2FA owner login; CSRF-protected consent; exact redirect binding; hashed random credentials; token/app revocation; read/write scope enforcement; token-bound staged media; transactional publication; upload MIME/size checks; rate budgets; unsupported Pixelix features hidden or returned as empty JSON; signed packages and SMACKBACK baselines.
Disclosure	No exploitation is known. The live test created one intended post. Findings A-D require an owner-authorized or stolen client credential except unauthenticated registration residue in Finding A. Finding E is a release reproducibility and operator-assurance issue.

1. Executive result

Pixelix successfully completed the real device path that SECAUDIT 043 left as a release condition: dynamic registration, browser authorization, password login, 2FA, consent, token exchange, v2 media upload, ALT update, JSON status creation, profile refresh, thumbnail retrieval and display. The server returned 200 for status creation and the resulting photograph appeared in both SNAPSMACK and Pixelix. The final deployed adapter returned valid JSON from public instance discovery and produced no new PHP error.

The interoperability pass also exposed missing optional API routes. Settings, the Pixelfed account alias, city search and collections initially returned HTML 404 pages. The final tag routes those calls through the adapter, advertises unsupported collections/groups/stories as hidden, and returns honest empty JSON for the unsupported location and collection searches.

The successful contract test exposed five follow-up issues. All are now closed: bounded lifecycle maintenance removes stale draft media and OAuth residue; client status reads are published-only; missing token expiry fails closed and legacy rows are revoked; authorization codes and refresh tokens use locked, conditional single-winner updates; and Smack Central permanently records and refuses reuse of every published identifier.

2. Live evidence

```
POST /api/v1/apps 200
GET /oauth/authorize 302 -> login/2FA
POST /oauth/authorize 302 -> native callback
POST /oauth/token 200
GET /api/v1/accounts/verify_credentials 200
POST /api/v2/media 202
PUT /api/v1/media/1 200
POST /api/v1/statuses 200
GET /api/v1/accounts/1/statuses 200
GET /img_uploads/.../thumbs/...jpg 200
```

Before the final compatibility repack, the access log also recorded 404 for `/api/pixelfed/v1/web/settings`, `/api/pixelfed/v1/accounts/1`, `/api/v1.1/compose/search/location`, and `/api/v1.1/collections/accounts/1`. The final tag contains explicit routes and JSON responses for those calls. A read-only live request to `/api/v1/instance` after repacking returned 200 JSON with `status_count: 1`.

3. Finding A - abandoned media, registrations and limiter rows never expire (MEDIUM, CLOSED)

Client registration inserts a permanent `snap_oauth_apps` row before the owner has consented. Media upload writes the original image, generated thumbnails, a draft `snap_images` row and `snap_oauth_media` ownership before a post exists. The rate limiter adds a permanent row for each source-address hash. There is no scheduled or request-bounded cleanup for an app that never receives a token, an expired authorization-code row, an abandoned Pixelix draft, its files and thumbnails, or inactive limiter buckets.

The 300-image/hour authoring budget and ten-registration/hour source budget reduce burst rate but do not bound cumulative storage. At their stated ceilings, a lost or compromised authorized client can create thousands of durable draft images per day. Unauthenticated sources can also grow registration records over time. The CMS visibly retaining an abandoned upload as a draft confirms this is the implemented lifecycle, not a theoretical path.

Remediation. `core/pixelix-lifecycle.php` defines a fixed seven-day retention period for unpublished client media, one day for unused registrations and expired authorization rows, two hours for inactive limiter buckets, and 30 days for expired or revoked credential history. Every adapter request runs one bounded pass. Draft image rows are locked before their ownership and database rows are removed, so concurrent publication wins safely. Original and generated thumbnail paths pass the shared `img_uploads` containment validator and a second resolved-root check before unlinking. `tools/pixelix-cleanup.php --dry-run` reports up to 100 candidates without changing them.

4. Finding B - status detail can disclose unpublished posts to a read token (MEDIUM, CLOSED)

`GET /api/v1/statuses/{id}` authenticates a read-scoped bearer and calls `px_status()`. That helper selects `snap_posts` by numeric ID with no `status='published'` predicate and no OAuth ownership predicate. A stolen or over-broad read token can enumerate IDs and retrieve draft or scheduled post captions, content warnings and attached media that do not appear in the public timeline.

The endpoint is an owner-client surface, but OAuth credentials are deliberately weaker and more portable than the authenticated administration session. Read scope should mean the public/profile client view, not implicit access to CMS drafts from every authoring path.

Remediation. `px_status()` now selects only `status='published'`. That rule applies to direct detail, timelines and the status returned immediately after publication. Draft and scheduled numeric IDs therefore resolve as absent to the client while published posts remain readable. The compatibility regression pins the predicate.

5. Finding C - null access-token expiry fails open (MEDIUM, CLOSED)

Bearer lookup accepts a token when `token_expires_at IS NULL OR token_expires_at > NOW()`. New code exchanges set a 30-day deadline, but a partially migrated, manually restored, legacy or damaged row with a valid token hash and null expiry becomes non-expiring. This contradicts the stated 30-day access-token lifetime and uses missing security metadata as permission.

Remediation. Bearer lookup now requires `token_expires_at > NOW()` with no null exception. The bounded lifecycle pass revokes existing bearer rows whose expiry is null. Regression coverage rejects reintroduction of the previous fail-open predicate.

6. Finding D - code exchange and refresh rotation are not atomic (LOW, CLOSED)

Authorization-code exchange selects an unused code and then updates the token row in a separate statement. Refresh selects the current refresh hash and then rotates it in a separate statement. Neither update includes the old hash/state in its `WHERE` clause, checks affected-row count, nor locks the row in a transaction. Concurrent redemption can therefore let two requests pass the initial check. Both can receive apparent success, while only the last stored token/refresh hash remains valid.

This is primarily a reliability and replay-hardening defect rather than a durable privilege escalation: the overwritten credential stops working. It can still produce confusing authentication failures and weakens the single-use and rotation guarantees under concurrency.

Remediation. Both grant paths now start a transaction and select the exact credential row `FOR UPDATE`. The update repeats the old hash, expiry, null-token or revocation condition and succeeds only when exactly one row changes. A stale or losing request receives `invalid_grant`; the refresh lifetime remains the original absolute 90-day deadline rather than being extended by rotation. Regression coverage pins the lock, conditional compare-and-swap and affected-row requirement.

7. Finding E - the published 520D tag was mutable after deployment (MEDIUM, CLOSED PROCESS)

`v0.7.520D` was published, deployed, force-moved to repair the database bootstrap path, deployed again, then force-moved again to add optional Pixelix routes. The final tag resolves to `48099d45`, but earlier archives with the same public version string contained different bytes and behavior.

SMACKBACK and package signatures protect the archive an operator actually installs. They do not make a moved tag reproducible, prevent GitHub/CDN archive caching from serving stale bytes, or let two operators determine which 520D they received from the version label alone. Reusing a release identifier also complicates incident response and audit correlation.

Remediation. 520D is retired and must not move again. The guarded Git release flow already refuses an existing local or fetched remote tag. Smack Central now adds a second, independent publication boundary: before either distribution is published it burns the version into a permanent `release-identifiers.json` ledger containing the track, tag, exact source commit, artifact SHA-256, signature and publication time. An existing ledger entry, archive or database history row refuses the build. Deleting an archive or rotating short database history cannot make the identifier reusable. Failed builds consume their identifier and corrections receive the next version.

8. Compatibility fallbacks review

- Settings, account alias, location search and collections fallback all sit

behind bearer authentication and require read scope.

- Returning an empty location list is preferable to inventing place metadata or

sending search terms to an undisclosed third-party geocoder.

- Collections, Groups and Stories are advertised as hidden. The empty

collections response remains a safe fallback for a client with stale UI state.

- Rewrites are present in both the updater template and the installer's separate

fresh-install rules. The release builder rejects a tag missing the primary adapter files/routes.

- Unsupported operations return JSON 501 after bearer authentication rather

than falling through to an HTML application page.

9. OAuth and authoring controls retained from SECAUDIT 043

- The site-mode gate fails closed outside GRAMOFSMACK.
- The owner Offline Posting switch gates registration, consent, upload, ALT changes and publication.
- Consent uses the normal owner session, password/2FA path and global CSRF autovalidation.
- Client ID/secret, authorization code, access token and refresh token are generated from cryptographic randomness; stored secret material is hashed.
- Exact registered redirect matching prevents substitution during consent and exchange.
- Staged media ownership is bound to the OAuth token and publication locks the rows in a transaction.
- Read and write scopes are explicitly checked.
- Registration and authoring budgets are serialized with database row locks.
- Uploads accept only JPEG, PNG and WebP detected from file content and cap input size at 64 MiB.
- Absolute refresh expiry, atomic rotation and administrative revocation remain in place.
- Pixelix does not currently send PKCE or OAuth state in the observed native contract. That inherited compatibility limitation remains documented and is not being represented as a modern PKCE-native OAuth profile.

10. Verification performed

- Reviewed the exact final `v0.7.520D` commit (`48099d45`) rather than relying on the later working branch.
- Reconciled Pixelix 5.0.0 source endpoints and required response models with Apache access logs from the real Android test.
- Traced registration, browser/2FA return, consent CSRF, token exchange, refresh, revocation, bearer authentication, media storage, ALT mutation, publication transaction, status/profile reads and optional compatibility routes.
- Confirmed the final tag contains the database-bootstrap repair, API/OAuth rewrite rules, fresh-install parity and the compatibility regression gate.
- Ran PHP syntax checks, the Pixelix compatibility regression and whitespace checks. They passed.
- The repository-wide canonical EOF/truncation scanner passed before the final tag repack: 1,023 tracked source files checked, zero failures.

- Added lifecycle, published-status, null-expiry, locked exchange and immutable

release-ledger regression gates. PHP syntax, compatibility, release workflow, whitespace and canonical EOF checks pass.

- No destructive disk-exhaustion test was performed against the live site.

11. Release decision

The narrow owner-authorized posting path is operational. The original six findings from SECAUDIT 043 and all five live follow-up findings in this report are remediated. The adapter is suitable for continued Pixelix testing and promotion on GRAMOFSMACK installations whose owner enables Offline Posting. The credential remains a public-posting capability and should be revoked when a device is lost or retired.

This is an internal AI-assisted security review, not independent certification or a penetration test.