SMACKVERSE Piggyback Search — Security Audit

Delta Review — v0.7.373–0.7.376 · July 4, 2026 · Audit 032

Executive Summary

This audit reviews the SMACKVERSE additions shipped across v0.7.373–0.7.376: the multi-account **piggyback authenticated search** feature (a read-only OAuth token stored on the blog to proxy a trusted instance's `/api/v2/search`), its encrypted token store and admin management UI, the `/api/v2/search` proxy, the client-side rendering of remote search results, and the related apostrophe-encoding fix on the public `/zuckit` profile and ActivityPub actor document. Scope covers `core/smackverse.php`, `smack-smackverse.php`, `smack-pixelfed.php`, `assets/js/ss-pixelfed-client.js` and `core/public-profile.php`.

The audit identified **1 medium finding** (piggyback token key fell back to a shared public default salt — **fixed in v0.7.376**) and reviewed 6 further items, all of which pass. No authentication bypass, remote code execution, privilege escalation, SSRF, or stored-XSS vulnerabilities were found. The new surface — storing a third-party credential and proxying an authenticated search — is server-side only, the token is never sent to the browser, the proxy is SSRF-guarded and admin-gated, and adding a token is protected by step-up (password + 2FA) authentication.

#	FILE	SEVERITY	FINDING	STATUS
1	`core/smackverse.php` `sv_search_salt()`	MEDIUM	Piggyback token key fell back to the shared public default salt when `download_salt` was unset — "encrypted at rest" degraded to obfuscation	FIXED
2	`smack-smackverse.php` `delete_search_account`	LOW	CSRF on token delete — reviewed; covered by the global auto-validator	PASS
3	`core/smackverse.php` `sv_authed_get()`	INFO	Bearer token sent to an admin-configured host; SSRF-guarded and host-validated	PASS
4	`ss-pixelfed-client.js` `renderResults()`	INFO	Remote search results rendered via `textContent` (no HTML injection)	PASS
5	`core/smackverse.php` `sv_search_token_*`	INFO	AES-256-CBC without HMAC (unauthenticated) — acceptable for at-rest	PASS
6	`smack-smackverse.php` `add_search_account`	INFO	Adding a token is step-up gated (password + 2FA); token field is `type=password`	PASS
7	`core/public-profile.php`	INFO	Apostrophe fix decodes then escapes / JSON-encodes — no injection	PASS

Findings

1. Piggyback Token Key Fell Back to Shared Public Default Salt

MEDIUM FIXED

core/smackverse.php · sv_search_salt() / sv_search_token_encrypt()

Piggyback search tokens are stored encrypted (AES-256-CBC) with the key derived from the site's `download_salt`. On installs that never set a `download_salt`, the key material fell back to the codebase-wide default constant:

```
return $s !== '' ? $s : 'snapsmack-default-salt-change-me';
```

Because that default is a fixed string present in the public source, any install relying on it encrypts every token under a key an attacker already knows. "Encrypted at rest" then provides no more protection than base64 obfuscation: anyone able to read the `snap_ap_search_accounts` table (a DB dump, backup leak, or SQL injection elsewhere) can recover the third-party access token. The token grants read access to the configured instance under the blog's chosen account. (The same shared-default weakness applies to FTP-password storage, which keys off the same salt — noted for a future pass.)

Fix (shipped, v0.7.376): `sv_search_salt()` now generates a random 32-byte per-install secret on first use and persists it as `smackverse_search_key`, independent of `download_salt`. The dedicated key is authoritative and stable, so a stored token is genuinely encrypted and rotating `download_salt` never orphans it. The shared default is no longer used. No stored tokens were affected (the feature is new and the earlier save path was failing).

Items Reviewed — No New Risk Found

CSRF on Token Delete

LOW · PASS · smack-smackverse.php delete_search_account

The per-row REMOVE control posts `action=delete_search_account` without an explicit token check in the handler. Verified this is covered by the framework: `core/auth-smack.php` calls `csrf_check()` on every admin request, and `core/admin-footer.php` auto-injects a hidden `csrf_token` into every `<form method="post">` (the inject regex is case-insensitive and matches the lowercase form). So delete — and every other POST handler on the page — is CSRF-protected without per-handler code. Adding a token additionally requires step-up (password + 2FA) reauth. No new risk.

Authenticated Search Proxy / SSRF

INFO · PASS · core/smackverse.php sv_authed_get() / sv_authed_search()

The proxy issues a Bearer-authenticated GET to `https://<host>/api/v2/search`. The host is not per-request user input — it is the fixed instance stored with the account (regex-validated on entry) — so the endpoint cannot be redirected at an arbitrary target by a caller. The request is guarded by `sv_resolve_public()`, the same SSRF/DNS-rebinding guard used for inbound federation, which pins the vetted public IP and refuses private ranges. Response size is capped at 1 MB and timeouts are bounded. The proxy is only reachable through the admin-authenticated SMACKVERSE client. No new risk.

Rendering of Remote Search Results

INFO · PASS · assets/js/ss-pixelfed-client.js renderResults()

Account names, handles and post captions returned by the remote instance are rendered through the client's `el(tag, cls, txt)` helper, which assigns `element.textContent` — never `innerHTML` — so remote-controlled strings cannot inject markup or script into the admin DOM. Remote image URLs are assigned to `img.src` (which does not execute script). No stored-XSS path. No new risk.

Token Storage Cipher & Handling

INFO · PASS · core/smackverse.php sv_search_token_encrypt/decrypt

Tokens use AES-256-CBC with a random per-message IV, without an HMAC (unauthenticated encryption). For at-rest storage this is acceptable: a tampered ciphertext only yields a broken token and a failed API call, and no decryption oracle is exposed to any party who cannot already write the database. The token is never returned to the browser (`sv_list_search_accounts()` selects host/username only) and the admin input field is `type=password`. No new risk.

Apostrophe Decode on /zuckit and the Actor Document

INFO · PASS · core/public-profile.php / actor doc

The double-encoding fix `html_entity_decode()`s the blog name/tagline/bio on load, then escapes on output via `pp_e()` (`htmlspecialchars`, `ENT_QUOTES`) on the HTML page and `json_encode` in the actor document. Decode-then-escape keeps a single, correct encoding — no XSS is introduced. No new risk.

Overall Assessment

The piggyback feature genuinely expands SMACKVERSE's attack surface: for the first time the blog stores a third-party credential and makes authenticated outbound calls on the operator's behalf. That surface is handled conservatively — the token is server-side only and never reaches the browser, the outbound proxy reuses the established SSRF guard against a fixed validated host, adding a token requires step-up authentication, and the client renders untrusted remote data through text nodes only.

The one substantive issue was the encryption key falling back to a shared public constant, which is now fixed with a dedicated per-install key (v0.7.376). With that shipped there are **no open findings** in this delta. A recommended future item, out of scope here, is to give FTP-password storage the same dedicated-key treatment, since it shares the old `download_salt` fallback.

FINDING	FILE	SEVERITY	STATUS
Token key fell back to shared public default salt	`core/smackverse.php`	MEDIUM	FIXED
CSRF on token delete (covered by global validator)	`smack-smackverse.php`	LOW	PASS
SSRF on authenticated search proxy	`core/smackverse.php`	INFO	PASS
XSS via remote search results	`ss-pixelfed-client.js`	INFO	PASS