Danny Wallet

Security Audit Report

Scope	Browser wallet (app.dannywallet.com) · Desktop
Network	Danny Chain (EVM, chainId 5069)
Codebase	danny-wallet — Next.js 14, ethers v6, WalletConnect v2
Audit type	Internal source review (white-box)
Date / Version	June 25, 2026 (rev. 1.1) · v1.1

OVERALL RATING

● Good

No critical, fund-loss vulnerabilities found

0	1	1	2
Critical	High	Medium	Low / Info

Executive Summary

Danny Wallet is a **non-custodial** wallet — private keys and recovery phrases are generated and stored only on the user's device, encrypted with a PIN. No server holds funds or can access keys. The core architecture is strong and follows good practice: WebCrypto (PBKDF2-SHA256, 210,000 iterations + AES-256-GCM), PIN brute-force protection, no logging of secrets, and a PIN is required for every signing action. This review found **one high-severity issue** (Permit phishing via typed-data signing) which has been **fixed and deployed**, plus three lower-priority hardening observations.

Reviewed Areas

✔ **Cryptography & key management**
PBKDF2-SHA256 (210k iters) + AES-256-GCM; random salt/IV per blob; GCM-tag PIN check

✔ **PIN brute-force protection**
Exponential cooldown after 5 fails; wallet wipe after 10

✔ **Transaction & message signing**
PIN required every time, no auto-sign; ERC-20 calldata decoded & warned

✔ **dApp browser (iframe)**
Cross-origin + no provider injection — dApp cannot reach wallet keys

✔ **Data storage & privacy**
Only encrypted blobs stored; no plaintext key/seed; no tracking; clipboard auto-clear 30s

✔ **Network & headers**
HTTPS; CSP frame-ancestors; nosniff; referrer-policy

Findings

ID	Sev.	Finding	Status
HIGH-01	High	Permit / typed-data blind signing — content not shown; risk of signing a Permit (gasless approve) that drains tokens	✔ Fixed

ID	Sev.	Finding	Status
MED-01	**Medium**	Wallet wipe after 10 wrong PINs — seedless imported accounts without a key backup risk permanent loss	**Risk accepted**
LOW-01	**Low**	Dependency hardening — removed @stablelib/ed25519 and patched elliptic via package overrides (npm prod criticals now 0)	**✔ Fixed**
LOW-02	**Low**	DappBrowser broad sandbox — mitigated by cross-origin	**Acknowledged**

HIGH-01 — Fixed. The WalletConnect confirmation screen now decodes typed-data requests and shows what is being signed. When a request is an EIP-2612 **Permit** (a gasless, signature-based token approval), it displays a strong warning including the spender and amount — preventing blind-signed approvals that could drain tokens. Status: fixed, build/tsc passing, deployed to production.

Recommendations

1. Prompt for key backup on import/create of seedless accounts (mitigates MED-01).
2. Add a second confirmation step for unlimited token approvals.
3. Offer an optional longer PIN / passphrase to reduce offline brute-force risk.
4. Plan a WalletConnect upgrade to close LOW-01 once a stable version is available.
5. Add a stricter CSP script-src for defense-in-depth against XSS (long term).
6. Consider an external audit by an independent firm before broad / high-value usage.

Remediations (rev. 1.1)

✓ Dependency overrides removed @stablelib/ed25519 and patched elliptic — npm production criticals reduced to 0.

✓ Unlimited token approvals (approve max / Permit) now require an explicit acknowledgement checkbox before signing.

✓ Import / create flows now warn that seedless accounts are not in the recovery phrase and must be backed up.

Disclaimer. This is an internal source-code review, not an audit by an independent third-party firm. It covers the wallet client only — not DEX/bridge smart contracts or network infrastructure. The report reflects the state at the audit date; re-audit when significant crypto, signing, or dependency changes are made.