

BitTool Signer

iOS App Version 1.0.0 Code Security Audit

Latest working-tree reassessment | Revision R3

ASSESSMENT: NO OPEN HIGH OR CRITICAL FINDING

The previous Safe High and both Low findings are closed for the identified cases. One new Low finding affects Bitcoin result/history summaries. No private-key extraction, weak-randomness fallback or password-authorization bypass was established.

0

CRITICAL

0

HIGH

0

MEDIUM

1

LOW

Assessment identity	Value
Product / build	BitTool Signer for iOS / app version 1.0.0 / build 1
Baseline	Frozen working tree: 59 modified files and 1 new file relative to the base commit
Capture	18 October 2026, 10:43:59 UTC / 21:43:59 Sydney
Report ID	BTS-SEC-2026-1018-R3
Prepared by	GPT-6 Astra
Method	AI-assisted source review, local tests, build and dependency checks

Prepared for the project owner. This independent AI-assisted assessment is not an audit-firm certification, human sign-off, guarantee of safety or proof of absence of vulnerabilities. Conclusions apply only to the source snapshot and evidence boundaries identified in this report.

01 / DECISION AND MATERIAL RISK

Executive assessment

The principal R2 security defects have been remediated

Safe recursive native outflows and ignored delegate values are now handled by the corresponding code paths and regression tests. Bitcoin high-fee classification accounts for verified change. Solana burn summaries aggregate matching assets and avoid a misleading scalar amount for mixed effects.

Current conclusion

No Critical, High or Medium finding was established in this reassessment. One Low finding remains: a Bitcoin transaction containing both verified change and a nonzero OP_RETURN output can produce a false self-return summary. The signing card still clearly identifies the burn, amount and High risk; the defect affects the result and retained activity record.

The new recovery-based password reset checks the derived wallet identity before rewriting the vault. Focused tests reject a different phrase, a different BIP39 passphrase and stale authorization. No wrong-wallet overwrite or password bypass was demonstrated. A stale lookup presentation race is documented separately as a reliability observation.

Wallet custody and randomness

Mnemonic creation requests 128-256 bits from the native system secure-random interface. Error returns stop generation; no time-based, identifier-based or weak PRNG fallback was found. This is a cryptographically secure random-source design under the trusted-platform assumption, not an unconditional promise of physical true randomness on every device.

Release recommendation

Fix the Low receipt/history defect and the stale lookup presentation before release, then rerun the affected tests. Retain the existing High-risk consent, full-detail review and wallet-password requirements. For substantial real-funds use, obtain independent human review and physical-device/distribution validation; this source assessment does not certify those properties.

Evidence overview

210 engine tests passed; 177 native/UI logical tests passed; 1,509 bounded protocol mutation cases produced no recorded anomaly. The JavaScript bundle rebuilt byte-for-byte, and an unsigned iOS Release build succeeded. Subsystem fixtures and evidence limitations are detailed later.

02 / REPORT GOVERNANCE

Document control and contents

Field	Record
Report ID / revision	BTS-SEC-2026-1018-R3 / final
Assessment date	18 October 2026 (Australia/Sydney)
Baseline capture	2026-10-18T10:43:59.281423+00:00
Change control	Working-tree snapshot; Git HEAD alone does not identify the audited code. See the manifest and patch.
Author	GPT-6 Astra; parallel subsystem review with consolidated findings
Distribution	Prepared for the project owner; no third-party delivery
Prior reports	Retained. Their findings apply to older snapshots; current closures are stated explicitly.

Section	Page
Executive assessment / document control	2-3
Scope, identity and threat model	4-5
Remediation verification	6
Open finding: Bitcoin result/history summary	7
Mnemonic randomness assurance	8
Native custody and recovery authorization	9
Chain, signing and transport coverage	10
Executed validation	11
Build integrity and supply chain	12
Observations and residual limits	13
Remediation and release criteria	14
Evidence and reproducibility	15
Primary references	16

Severity and confidence

Critical denotes established systemic key compromise or broad unauthorized loss; High denotes material asset risk or a core informed-consent failure; Medium denotes bounded security impact; Low denotes limited integrity or review inconsistency with material mitigating controls. Informational observations do not enter the open-finding count.

E1 = locally executed behavior; E2 = source/data-flow and primary protocol semantics; E3 = hardening or external assurance limitation. No CVSS score, CVE assignment or formal compliance certification is asserted for first-party findings.

03 / WHAT WAS ASSESSED

Scope and source identity

Scope	Coverage
Repository inventory	960 files; 38,870,920 bytes captured and hashed, including uncommitted code.
iOS application	51 Swift files: vault, BIP39, JavaScript bridge, authorization, setup, review, scanner, public storage and lifecycle.
JavaScript runtime	14 engine/src modules plus protocols.js: Bitcoin, EVM, TRON, Solana, NFTs, derivation, fixed signing plans and QR transport.
Tests and support	25 unit-test Swift files, 3 UI-test Swift files, 15 engine test files and 11 maintenance scripts.
Resources and packaging	Generated bundle, registries, wordlists, translations, assets, Xcode settings, privacy manifest, npm manifest and lockfile.

Immutable baseline identifiers

Base Git commit (not the complete working-tree identity)

3d706b7a80e30cd7121931f69fa79e0b68bc23a6

Source-SHA256SUMS.txt inventory SHA-256

51d5bf9a2a9ca5109ad745f8a07c672f8e861c1010d917ca86fcfadf11f0937e

Shipped JavaScript SHA-256

6933d0d8a6f60faa8222506ac0bd7eef80eb884c9f73e31e9f354b4740f21077

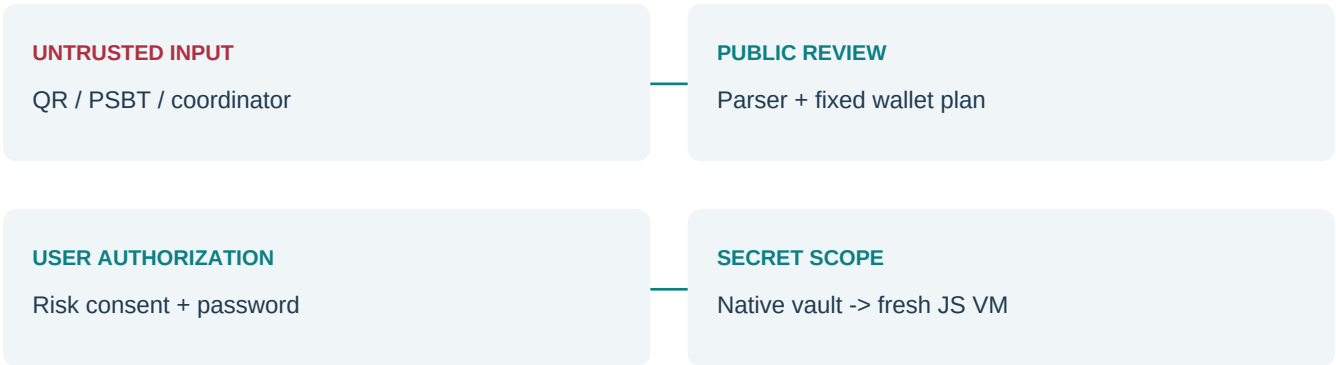
The snapshot includes 59 modified files and the new engine/test/audit-bts-2026-1018-r2.test.mjs. Tests and builds ran against the copied snapshot. The evidence package contains the file inventory, working-tree patch and new test file for reconstruction.

Scope qualification

First-party runtime code was reviewed by subsystem. Supporting resources and configuration received security-oriented and structural checks. Dependencies were inventoried, advisory-checked and traced into the bundle; their entire implementation was not independently audited. No numeric line/branch coverage claim is made.

04 / TRUST BOUNDARIES

Threat model and method



Asset / actor	Expectation
Recovery material	Mnemonic, BIP39 passphrase, seed, passwords and keys remain protected outside bounded authorized operations.
Untrusted coordinator	Controls QR/file/request bytes and unsigned labels. A claimed origin, address or fingerprint must not authorize a secret or replace a fixed signing task.
Signing user	Needs a faithful economic review as well as a signature over the intended bytes. High-risk acknowledgments do not excuse an inaccurate principal summary.
Local data corruption	Vault, public metadata and signed-plan integrity checks should reject tampering. A compromised OS remains outside the trusted-platform assumption.
Platform and distribution	Apple security APIs and app integrity are trusted. A local unsigned build does not prove App Store/TestFlight provenance.

Executed approach

The review combined full subsystem source reading, comparison with R2, existing regressions, harmless public-fixture checks, source/bundle comparison, finite parser mutation tests, a clean lockfile install, reproducible builds, native simulator tests and fresh dependency advisory queries.

OWASP MASVS control families [R1] provide a review lens for storage, cryptography, authentication, platform use, code quality and privacy. This does not constitute a MASVS conformance assessment. No live-chain transaction, funded wallet, production secret, broadcast or controlled EVM contract execution was used.

Excluded assurance

Physical-device fault injection, OS/hardware entropy validation, memory-dump/side-channel testing, exhaustive dependency review, third-party wallet interoperability certification, live contract state/bytecode, full linguistic review and formal verification remain outside this engagement.

05 / PRIOR FINDING DISPOSITION

Remediation verification

Prior item	R3 status	Current evidence
R2-01 / High Safe economic review	CLOSED for reported roots	MultiSend propagates nested forwards, prioritizes spending/delegating branch descriptions and explicitly reports omitted branches. execTransaction zeroes ignored delegate value. Seven relevant current regressions pass; static semantics match [R4/R5].
R2-02 / Low Bitcoin fee/change	CLOSED	Verified change is excluded from the fee-policy denominator. The public fixture with 0.001 BTC payment and 0.005 BTC fee now receives HIGH FEE/high with or without change.
R2-03 / Low Solana multi-burn	CLOSED for reported cases	Matching burn amounts aggregate. Mixed asset/transfer effects omit the misleading single amount and use a multi-operation headline; source/bundle controls pass.
R2 Info TRON history fee label	CLOSED by source review	ActivityTab uses Energy budget when feeKind is energyBudget, rather than calling it a total fee.
R2 Info Repair password lifetime	CLOSED by source review	The repair sheet clears unsubmitted password on securityEpoch changes when no operation is busy, and on dismissal.
Earlier Bitcoin burn-plan serialization	NO REGRESSION OBSERVED	20 fresh source/bundle PSBT v0/v2 burn controls resolve/sign and preserve unsigned bytes.
Earlier Solana single-burn review	NO REGRESSION OBSERVED	Six fresh source/bundle end-to-end fixture controls preserve the prominent burn and High risk.

Closure boundaries

Safe closure is based on source semantics and the existing current regressions, not on executing arbitrary Safe contracts on an EVM. The decoder remains bounded and cannot establish live code/state effects. The original root causes are closed; this is not a claim of universal contract simulation.

Earlier native fee-format and BIP39 restore-limit changes remain present and are covered by current source review/native suites. The previous round's separate 50-vector multilingual harness was not re-executed in R3 and is not counted as new evidence.

Code references: engine/src/evm.js:239-243, 270-321; resolver.js:154-157; solana.js:401-415; ActivityTab.swift:228-235; WalletSecuritySheets.swift:330-333

06 / LOW - OPEN - E1 + E2

R3-01 | Bitcoin history summary

A burn with verified change can be recorded as a self-return

The principal signing card is correct, but a separately computed summary discards data outputs and can falsely state that every output returns to the wallet.

Attribute	Assessment
Affected component	engine/src/resolver.js:161-163; result and activity consumers
Prerequisites	A valid Bitcoin signing plan with verified change and a nonzero OP_RETURN output; no ordinary external payment is needed.
Observed behavior	Source and shipped bundle produce summary: Every output returns to your wallet, while display.action is Burn, display.amount is 0.99 BTC and riskLevel is high.
Impact	Misleading signing-result and retained-history text; impairs later recognition/reconciliation of an irreversible burn.
Mitigation / severity	The preapproval card, exact burn amount and burn warning remain visible; risk acknowledgment and password are still required. No signature bypass or concealed preapproval burn was demonstrated. Low severity.

Root cause and propagation

withVerifiedChange computes sent using outputs that are neither data nor verified change. When the only remaining non-change output is OP_RETURN, sent is zero, selecting the self-return summary. The separate bitcoinDisplay receives all outputs and correctly prioritizes the burn.

SignatureResultView displays review.summary, and AppStore persists approved.summary into the signing receipt. ActivityTab consumes this summary in list/detail views; retained receipts do not preserve all transaction details. Thus a correct pre-sign display does not prevent misleading post-sign history.

Recommended correction and acceptance test

Derive the human summary from the same structured effects as the main display. Preserve burn amount/count and distinguish mixed burn, payment and change. Add source/bundle and receipt tests for burn+change, burn+payment+change, zero-valued OP_RETURN and change-only consolidation.

Code references: engine/src/resolver.js:151-169; SignatureResultView.swift:40-41; AppStore.swift:449-457; ActivityTab.swift:72, 82-86, 126-127, 157-160. Evidence: btc-solana/bitcoin-burn-retest-results.json and bitcoin-boundaries-results.json.

07 / WHAT THE CODE DOES AND DOES NOT GUARANTEE

Mnemonic randomness assurance

Appropriate secure-random design; no unconditional true-random guarantee

The current production path uses Apple's secure-random API, checks its status and fails closed. A source review cannot prove that every future OS, device, emulator or compromised runtime will always supply unpredictable physical entropy.

Generation path

Step	Observed control
Native request	AppStore.generateMnemonic calls SecureRandom.entropyHex. Supported word counts are 12, 15, 18, 21 and 24, corresponding to 16, 20, 24, 28 and 32 entropy bytes.
System source	SecureRandom.bytes uses SecRandomCopyBytes(kSecRandomDefault, ...); a non-success status throws randomFailure.
BIP39 conversion	keys.js accepts exactly 128-256 bits in 32-bit steps and passes the entropy to @scure/bip39 entropyToMnemonic. The checksum is not additional secret entropy.
JavaScript entropy	The native bridge installs a non-writable crypto.getRandomValues and native-backed Math.random before loading the bundle. Missing/incorrect-length bytes throw; no weak fallback was found.

CSPRNG versus physical randomness

Apple documents a system cryptographic PRNG seeded from multiple entropy sources, potentially including hardware TRNG output [R2/R3]. Applications consume the secure system interface, not a certified raw physical-random stream. Trust depends on the device, OS and app integrity; a compromised source can return success with predictable bytes.

Evidence quality

SecurityTests.nativeEntropyIsBoundedAndUnique checks 12/24-word output lengths, rejection of word count 13, byte bounds and differing samples. javascriptBridgeHasFreshScopeNativeEntropyAndNoNetwork checks native-backed output, fresh state, absent fetch/XMLHttpRequest and inert quoted input. These checks do not prove entropy quality or simulate a failing SecRandomCopyBytes status.

The OS-status fail-closed conclusion is static evidence. No hardware entropy audit, statistical certification, failure-injection run or absolute uniqueness guarantee is claimed. Imported recovery phrases inherit the entropy quality of their original generator.

Code references: Core/VaultCrypto.swift:47-62; Models/AppStore.swift:161-165; Core/SigningEngine.swift:25-62; engine/src/keys.js:16; SecurityTests.swift:22-31, 107-131.

08 / AUTHORIZATION AND SECRET LIFETIME

Native custody and recovery

Control	Assessment
Vault encryption	Nested AES-GCM password and device layers authenticate exact serialized header bytes. PBKDF2-HMAC-SHA256 uses 600,000 rounds; imported KDF/ciphertext sizes are bounded before expensive work.
Device key / storage	Device-bound Keychain protection, passcode requirement, bounded file reads and authenticated public metadata remain in place. Simulator/host checks do not attest physical device hardware.
Signing authorization	Public-index resolution produces fixed tasks. Signing revalidates request/plan/key identity; wallet password remains required. App Lock biometric access does not itself authorize signing.
Secret exposure	Secrets/untrusted requests use fresh JavaScript VMs. Public-only warm contexts and scan sessions are segregated. Error messages avoid raw secrets; memory clearing is best effort for Swift strings and managed JS objects.
Recovery password reset	A valid canonical phrase and exact BIP39 passphrase derive a walletKeyId; AppStore requires equality with the target public identity before calling the lower-level vault rewrite.

Recovery is an authorization credential

A holder of the correct recovery phrase and BIP39 passphrase already controls the wallet keys; requiring the forgotten old password is not necessary for this recovery design. A phrase without the correct nonempty passphrase derives a different wallet and is rejected. The low-level `WalletVault.resetPassword` intentionally relies on its AppStore caller for identity validation.

Independent recovery checks

A 12-case host harness using exact AppStore recovery methods, real BIP39 derivation, `VaultCrypto` and `WalletVault` passed. It rejects wrong-wallet phrases, wrong passphrases, invalid mnemonics, missing public identity, weak replacement passwords and suspension during derivation. Correct recovery preserves metadata/secret identity, invalidates the old password and rotates the security epoch.

Harness seams: fixture engine returns indexes freshly derived by this snapshot, Keychain is in memory, and `SwiftUI` is absent. The selected iOS UI test separately exercises successful in-place password reset. See the validation section for simulator results.

Code references: `Core/WalletVault.swift:173-186`; `Models/AppStore.swift:484-512`; `native/harness-provenance.json`; `native/reset-authorization-results.json`.

09 / SIGNING AND REVIEW SURFACES

Chain and transport coverage

Surface	Reviewed protections and boundaries
Bitcoin	PSBT v0/v2, prior-transaction commitments, scripts/policies, sighash restrictions, owned-input matching, verified change, fees, burns, metadata preservation, accumulated signatures and plan binding.
EVM / Safe / NFT	Transaction envelopes, chain IDs, personal/EIP-712 data, permissions, permits, Safe and MultiSend recursion, native value, NFT marketplace orders, collection-wide approvals and decode/risk classification. Unknown code/state is not simulated.
TRON	Protobuf field bounds, address conversion, contract dispatch, fee_limit as an energy budget, multisignature identity and incremental signatures. Offline network identity and live permission membership remain unverifiable.
Solana	Message structure, static account indexes, signer slots, transfers/burns/authority changes, SPL/Token-2022, stake, NFT/Squads classifications and signature preservation. Address lookup tables are rejected.
Public identity / resolver	Wallet metadata validation, deterministic key identifiers, fixed task derivation, transport/request/review hashes and re-derivation from the authorized seed.
QR and file input	Strict JSON/CBOR boundaries, duplicate-key and precision checks, UR/fountain state, BBQR decompression limits, payload/frame caps and transport binding.

Current bounded parser experiment

The 1,509-case deterministic corpus covers UR/CBOR, fountain fragments, BBQR, decompression boundaries, file limits and JSON. Source and bundled dispatch agree; accepted transport is revalidated; no recorded anomaly or uncaught dispatch exception occurred.

Some altered public payloads remain valid and are expected to be accepted. Synthetic PSBT framing controls need not be valid transactions; subsequent review rejections are counted separately. This finite corpus is not exhaustive or coverage-guided fuzzing.

Supporting resources

Twenty locales and 773 interface keys per locale passed key/nonempty/format-specifier checks. This does not establish that every security warning is linguistically correct. Local registries and names remain context, not proof of a live contract's behavior.

10 / RESULTS AND EVIDENCE LIMITS

Executed validation

Validation	R3 result	Qualification
Full JavaScript suite	210 / 210 passed	No failures or skips; includes 16 tests added since R2.
Native + selected UI	177 / 177 logical tests passed	189 parameter-expanded executions; 0 failed, 0 skipped.
Critical UI flows	6 selected tests passed	Create/backup, review/sign, password change, recovery reset and two scanner entry points.
Protocol mutations	1,509 cases; 0 anomalies	Fixed seed; source/bundle and transport checks.
BTC/SOL independent fixtures	108 case/implementation observations	72 successful signatures verified; 36 non-signing/negative observations; source and bundle counted separately.
Recovery authorization harness	12 / 12 checks passed	Exact relevant source methods with documented host seams.
Stale lookup harness	Race reproduced; reset rejected	Reliability observation; original vault unchanged.
Build / translation checks	Passed	Byte-identical bundle; unsigned Release; 20 locales / 773 keys.

Counting discipline

The EVM/TRON reviewer ran 112 existing tests and the BTC/Solana reviewer ran 83 existing tests. These overlap the 210-test full suite and are not added to it. The 108 independent observations include positive/negative controls; they are not 108 unique transactions. Native logical counts and parameter-expanded executions are reported separately.

Environment and run outcome

Node.js 20.20.2; iOS Simulator 26.4, iPhone 17 arm64; Xcode developer tools at /Applications/Xcode.app. Tests used the frozen working tree. The R3 native/UI invocation and xcresult passed on the first full run. Post-run diagnostic collection logged a simctl lookup error; no additional diagnostic archive is assumed. The R2 timing failure is historical.

Existing tests and local signatures support the cases exercised. They do not prove a mainnet economic result, absence of all race conditions, resistance to a compromised OS or interoperability with every external wallet.

11 / REPRODUCIBILITY AND DEPENDENCIES

Build integrity and supply chain

The shipped engine matches a clean locked rebuild

The 1,181,026-byte signer.js matches both an in-memory esbuild reconstruction and npm run build after npm ci --ignore-scripts. The Release app contains the same bundle.

JavaScript SHA-256

6933d0d8a6f60faa8222506ac0bd7eef80eb884c9f73e31e9f354b4740f21077

Check	Observed result
Lockfile installation	116 packages installed with lifecycle scripts disabled. No dependency upgrade or product-code change was performed.
Bundle graph	273 esbuild inputs; 233 files contribute output. No contributing module path matched the network-provider scan; the native JS bridge independently exposes no network API.
Production advisory query	npm audit --omit=dev: 0 reported vulnerabilities at query time.
Full dependency query	6 affected dependency entries: 3 Moderate and 3 Low; 0 High/Critical. These are package entries, not six distinct first-party wallet flaws.
Advisory reachability	The uuid [R6] and elliptic [R7] advisories affect transitive development registry dependencies; no vulnerable-function execution was established. The affected uuid/elliptic packages do not contribute to the generated runtime bundle.
Unsigned Release	Build succeeds for generic iOS. App version 1.0.0 / build 1 / com.bittool.signer. Six test fixture markers absent from executable; no test artifacts found in the app.

Interpretation and follow-up

A clean advisory query is a database observation, not proof that dependencies have no undisclosed defects. Review and update the development registry dependency chain in a controlled change, preserving interoperability fixtures and reproducible output. Do not apply a forced major-version upgrade solely to silence an audit count.

The graph scan is supportive evidence only: path names cannot prove absence of arbitrary network behavior. The source and JavaScriptCore capability boundary provide the stronger offline constraint. Distribution signature, notarization/App Store provenance and binary-to-source equivalence of a shipped release were not verified.

12 / INFORMATIONAL - NOT OPEN SECURITY FINDINGS

Observations and residual limits

O-01 | Stale existing-wallet lookup presentation

An asynchronous existing-wallet lookup can finish after the user replaces the imported phrase. The old result may populate duplicateOf, offering password reset for the previous wallet. A controlled-delay harness with the exact WalletSetupModel reproduced this presentation state.

AppStore independently re-derives identity and rejects the wrong-wallet reset; the original vault remains unchanged. The subsystem reviewer classified the UI defect as Low reliability; the consolidated security register treats it as Informational because no authorization, confidentiality or stored-wallet integrity bypass was shown.

Tie each lookup to the current phrase/passphrase revision or cancel/invalidate it on edits and navigation. Invalidate the lookup revision and apply nil for a current no-match result; add a delayed-result test. Keep the independent AppStore identity check.

Code references: Features/Setup/WalletSetupModel.swift:90-97, 150-159; Models/AppStore.swift:498-512; native/lookup-lifecycle-results.json.

O-02 | Bounded semantic decoding

Safe forwards is a decoder-specific native-transfer summary, not a universal execution simulator. Unknown delegate targets and truncated/unsupported effects require conspicuous incomplete/high-risk treatment. Live contract state, callbacks, account permissions and execution success remain unverified offline.

Solana unknown-token grouping can split singular/plural labels; Metaplex burn paths may keep explicit High/partial burn warnings without participating in every structured movement summary. These are useful consistency hardening targets. No new hidden fully-decoded burn or signing bypass was established in this review.

O-03 | Release assurance and memory limits

Managed Swift/JavaScript memory cannot guarantee complete zeroization. The physical device, OS RNG, biometric hardware and distributed binary were not penetration-tested. Maintain platform updates, protected backups and independent release provenance checks.

O-04 | Development dependency advisories

The six advisory-affected development dependency entries warrant maintenance even though the affected packages are absent from the generated wallet runtime. Track the dependency update separately from first-party wallet vulnerability counts.

13 / ACTIONABLE FOLLOW-UP

Remediation and release criteria

Priority / owner	Action	Acceptance evidence
Before release Wallet review/UI	Fix R3-01: unify Bitcoin structured effects and summary generation.	Burn+change retains burn amount/count in review, result and activity. Mixed payment/burn and change-only cases are explicit; source/bundle agree.
Before release Native setup	Fix O-01: invalidate asynchronous duplicate lookup when phrase/passphrase changes.	Old lookup cannot set duplicateOf for a new phrase. Wrong-wallet reset remains rejected even if UI state is stale.
Maintenance Dependency owner	Resolve development dependency advisories through compatible upgrades.	Fresh full/production audit, existing protocol fixtures and byte-reproducible bundle; no forced unreviewed major upgrade.
Release qualification Maintainer / reviewer	Freeze a release commit and validate the distributed artifact.	Commit/tag maps to the final source manifest; signed artifact resource hash and build settings are recorded.
Independent assurance Security reviewer	Review complex chain semantics and physical-device controls.	Human review, representative third-party coordinator tests, device lifecycle/privacy checks and documented offline limitations.

Preserve the successful controls

Keep native secure-random failure handling, authenticated vault/public data, password-based signing authorization, transport binding and fixed plan revalidation. Preserve explicit High-risk consent for burns, approvals, delegation and opaque requests. Do not weaken these controls to improve test convenience or remove warnings.

Retest and closure policy

A maintainer response is not a verified fix. Close R3-01 only after reviewing the final patch and retesting the source, bundled engine and receipt consumers. Record a new baseline hash whenever source or generated resources change. This report does not include a remediation implementation or sign-off on later code.

14 / AUDIT TRAIL

Evidence and reproducibility

Evidence	Purpose
source-manifest.json / source-SHA256SUMS.txt	960-file baseline, capture time, sizes and digests; includes dirty-tree status.
working-tree.patch + new-source/	Tracked delta against base commit plus the newly added regression test.
snapshot-delta.json / final-baseline-recheck.json	Change inventory versus R2 and close-out comparison of live source/snapshot hashes.
javascript-tests.log / subsystem notes	Full 210-test result and overlapping targeted subsystem suites; static analysis notes.
btc-solana/*results.json + scripts	Independent source/bundle observations, signature checks and the Low summary reproduction.
native/*harness* + results	Recovery authorization and delayed-lookup evidence with explicit harness seams.
protocol-mutation-results.json + script	1,509 finite parser/transport cases, group counts and observations.
native-ui-tests.log + xcresult summary	Native/UI results; full .xcresult remains in the local audit directory.
build-verification.json / release-inspection.json	Bundle equality, module graph, package reachability and Release inspection.
npm-audit-*.json / localization-validation.json	Timestamped dependency observations and translation structural checks.

Reproduction outline

Use the base commit from page 4, apply `working-tree.patch` and restore `new-source/` at repository-relative paths. Verify every file against the source manifest before installing dependencies. Run `npm ci --ignore-scripts` and `npm test` in engine, then the evidence scripts and Xcode test command recorded in the `logs/run` metadata.

The compact evidence archive excludes `node_modules`, build products, the copied source tree, rendered page images and large `.xcresult` bundles. Full local artifacts remain under `tmp/audit-2026-10-18-r3`. Public test seeds are test material only and must never receive funds.

The accompanying `SHA256SUMS` file identifies the final report and evidence archive. Hashes prove correspondence to these files, not their security or authorship independently.

15 / EXTERNAL SOURCES AND ASSURANCE BOUNDARIES

Primary references

R1 | OWASP Mobile Application Security Verification Standard

Review control families; no conformance claim.

<https://mas.owasp.org/MASVS/>

R2 | Apple Platform Security: Random number generation

System cryptographic PRNG and entropy-source model.

<https://support.apple.com/en-euro/guide/security/seca0c73a75b/web>

R3 | Apple Developer: Randomization Services

Platform randomization API reference.

<https://developer.apple.com/documentation/security/randomization-services>

R4 | Safe v1.4.1: Executor.sol

CALL supplies value; DELEGATECALL has no immediate value argument.

<https://raw.githubusercontent.com/safe-global/safe-smart-account/v1.4.1/contracts/base/Executor.sol>

R5 | Safe v1.4.1: MultiSend.sol

Packed operation semantics and delegate execution context.

<https://raw.githubusercontent.com/safe-global/safe-smart-account/v1.4.1/contracts/libraries/MultiSend.sol>

R6 | GitHub Advisory: GHSA-w5hq-g745-h8pq

uuid output-buffer bounds advisory; development dependency context.

<https://github.com/advisories/GHSA-w5hq-g745-h8pq>

R7 | GitHub Advisory: GHSA-848j-6mx2-7j84

elliptic risky cryptographic primitive advisory; development dependency context.

<https://github.com/advisories/GHSA-848j-6mx2-7j84>

References were consulted on 18 October 2026. Safe v1.4.1 is a semantic reference, not evidence that any supplied on-chain target runs that code. Current registry/advisory results are included separately in the evidence archive.

Professional format; explicitly bounded assurance

Prepared by GPT-6 Astra. The report describes what was reviewed, what was observed, what remains open and what was not established. It does not represent a third-party audit company, legal attestation or guarantee against asset loss.