Nought Security Review

Final point-in-time report

Report date: 2026-09-10 Initial review commit: `e3211ecaf44df1ae64ae6471353f0a3f4514a669` Final reviewed commit: `e2d2c2d7eb9c0cb59d3827d452d38d5feecf02f4` Repository: Nought / `Robinhood` working tree supplied by the project Reviewer credit: Independent Robinhood contributor review Technical assistance: OpenAI Codex, used as an AI-assisted source-review and analysis tool

Important disclosure

This was an iterative security review conducted by an independent Robinhood contributor with AI assistance from OpenAI Codex. Codex is an AI coding agent, not a human auditor, audit firm, or certification authority. This report is not an endorsement by Robinhood or OpenAI, is not a warranty that the system is free of defects, and must not be represented as a formal security certification. The integrity hashes distributed with this report prove file identity only; they are not a cryptographic signature or statement of authorship.

Executive conclusion

The reviewed Nought system is a Groth16/BN254 shielded pool for multiple assets on an EVM chain. It uses a 2-input/2-output join-split circuit, Poseidon commitments and nullifiers, a rotating Merkle tree, relayer infrastructure, an API/indexer, a browser client, and a staged deployment and activation process.

At the final reviewed commit, no unresolved Critical, High, Medium, Low, or Informational implementation finding from this engagement remained open. The last item, M-34, was closed by making API synchronization evidence mandatory and bound to a specific chain block hash, and by maintaining the indexer's cursor and checkpoint atomically. Targeted read-only checks of that remediation found no new actionable issue.

One previously disclosed design risk remains deliberately accepted: nullifiers are not bound to a tree identifier. An identical commitment inserted at the same leaf position in two trees produces a colliding nullifier and can make the second note unspendable after the first is spent. Honest accidental collision is dominated by the 248-bit note blinding value, but the risk is not literally impossible and its consequences are permanent. The project accepted this trade to avoid an additional persistent write per commitment. It is recorded as AR-1 below and prevents the report from meaning "risk-free."

The review also confirmed that `Num2Bits(levels)` is the load-bearing constraint that makes `pathIndices` canonical. Because the Merkle path consumes only the low `levels` bits while the nullifier consumes the full field element, omitting this range constraint would allow one note to generate multiple nullifiers by adding multiples of `2^levels`. In the reviewed circuit, the constraint is present and connected.

System and threat model

The review treated the following properties as security-critical:

- circuit soundness and canonical public inputs;
- conservation of value per public `assetId` across every 2-in/2-out transaction;
- ownership and authorization of note spends;
- uniqueness and durable tracking of nullifiers;

- continued spendability across Merkle-tree rotation;
- exact token and native-asset accounting, including non-standard tokens;
- consistency among proving artifacts, the generated Solidity verifier, and deployed runtime bytecode;
- safe staged launch, governance handover, deposit activation, and relayer readiness;
- reorganization-safe note and nullifier indexing;
- fail-closed browser, API, relayer, and release behavior;
- integrity and browser usability of large proving artifacts served through mirrors.

Adversaries considered included malicious proof generators, depositors, recipients, relayers, API callers, token contracts, listing applicants, mirror/CDN operators, front-runners, chain reorganizations, configuration mistakes, stale or divergent duplicated code, and compromised short-lived deployment credentials.

Scope

Core protocol

- `contracts/ShieldedPool.sol`
- `contracts/RelayerRegistry.sol`
- `contracts/MerkleTreeWithHistory.sol`
- `contracts/ProtocolFee.sol`
- `contracts/StockSwapAdapter.sol`
- `contracts/interfaces/*.sol`
- `circuits/transaction.circom`
- `circuits/merkleProof.circom`
- `circuits/keypair.circom`
- generated verifier correspondence and proving-key/transcript checks used by the deployment path

`StockSwapAdapter.sol` was reviewed as source code and defects in it were remediated, but it was not part of the live pool deployment evidence.

Expanded integration and operational scope

The engagement expanded beyond the original approximately 900 nSLOC core because exploitable security boundaries crossed package and deployment boundaries. The expanded review covered the security-relevant portions of:

- `src/` client proof construction, note recovery, receipt verification, ABI declarations, artifact loading, and relayer interaction;
- `server/src/` routes, listing service, indexer, reorganization repair, API synchronization, and persistence;
- `relayer/server.js` and `relayer/standalone/server.js`;
- `app/` activation state, relayer and proving-artifact consumers, mirror fallback, and security-relevant build configuration;
- `scripts/` deployment, ceremony verification, governance handover, asset admission, live checks, readiness, mirror verification, live-flow tests, and operational watchdogs;
- security regressions and boundary tests in `test/`, `server/test/`, and `app` test sources;
- launch and operational documentation where incorrect commands or claims could cause an unsafe deployment.

Exclusions and limits of scope

The following were not independently audited as complete systems:

- third-party dependencies, Solidity compiler internals, circom, snarkjs, Groth16, BN254, Poseidon, RPC nodes, the Robinhood chain consensus, browsers, operating systems, CDNs, Vercel, and token contracts;
- custody and physical security of deployment, governance, fee-recipient, issuer, listing, and relayer keys;
- correctness of the multi-party ceremony beyond the repository transcript, hashes, final-key naming, source export, and runtime correspondence gates reviewed here;
- production-host hardening, penetration testing, denial-of-service capacity testing, privacy traffic analysis, and economic modeling;
- a formal mathematical proof of the circuits or contracts;
- code introduced after the final reviewed commit.

Generated artifacts and vendored/build output were not line-by-line source audited. They were considered only where hashes, sizes, CORS behavior, verifier export, or runtime correspondence were security properties.

Method

The review combined:

- manual line-by-line source and constraint review of the core contracts and circuits;
- data-flow review from circuit witnesses and public inputs through Solidity verification and state changes;
- state-machine and invariant review of tree rotation, roots, nullifiers, activation, governance, admission, relaying, listing, and indexing;
- adversarial review of duplicated implementations, stale caches, missing values, negative caches, error swallowing, partial failures, and unsafe defaults;
- review of tests for false positives, vacuous assertions, caller/boundary mismatches, and claims broader than the test actually established;
- mutation-style validation described in remediation evidence, where reintroducing a defect was expected to make the relevant regression fail;
- targeted read-only syntax, module, predicate, diff, and repository-state checks during closure rounds;
- point-in-time review of live RPC, API, relayer, verifier, ceremony, and readiness evidence supplied or exposed by the project.

At the user's request, full test suites were not repeatedly rerun during every later review round. Final suite counts and certain end-to-end results in project replies are project-reported evidence, not an independent laboratory reproduction. Targeted checks were used for the final remediation. This distinction is material.

Severity model

- Critical: direct and broadly exploitable theft, arbitrary minting, or systemic loss of control.
- High: realistic loss or permanent lock of funds, proof/deployment integrity failure, or defeat of a central trust boundary.
- Medium: meaningful security, availability, launch, indexing, or integrity failure requiring conditions or limited scope.
- Low: defense-in-depth, operational, privacy, diagnostics, or narrower correctness weakness.
- Informational: test-quality, documentation, maintainability, or assurance issue without a direct exploit on its own.
- Accepted Risk: understood design trade retained after challenge; not a claim that the risk is fixed.

Findings register - core and initial review

ID	Sev.	Finding	Resolution
C-01	Critical	Proving artifacts were not originally bound strongly enough to the reviewed ceremony and deployed verifier.	Fixed through transcript, final-zkey hash, verifier export, compiled artifact, deployed runtime, and real-proof gates.
H-01	High	Filling the active tree could prevent legitimate balances from remaining usable.	Fixed; sealed roots remain permanently valid and rotation behavior is covered.
H-02	High	A short rolling-root window could be cheaply flushed, invalidating otherwise valid spends.	Fixed by the sealed-root/rotation model and permanent final-root validity.
H-03	High	A reverting native-asset fee recipient could brick withdrawals.	Fixed by separating fee accounting/collection from user withdrawal completion.
H-04	High	`StockSwapAdapter` could spend unrelated balance held by the adapter.	Fixed; adapter was not part of the reviewed live deployment.
H-05	High	Receipt verification did not enforce the signed root in every client path.	Fixed across client implementations and backed by positive and reason-specific negative controls.
M-CORE-01	Medium	Token transfer behavior could violate pool accounting, especially on outbound fee-on-transfer or otherwise non-standard behavior.	Fixed with exact inbound and outbound balance checks; unsupported tokens fail rather than socialize loss.
M-CORE-02	Medium	`assetId` lacked an explicit field/range boundary.	Fixed by constraining/validating the public asset identifier.
M-CORE-03	Medium	Note feed/indexing assumptions treated a leaf index as globally unique and broke after rotation.	Fixed by carrying tree identity through commitment events and indexing.
M-CORE-04	Medium	Relayer receipt durability and replay/state assumptions were insufficient.	Fixed in the receipt/sign-off flow.
M-CORE-05	Medium	Relayer rate limiting and abuse resistance were incomplete.	Fixed in relayer handling.
M-CORE-06	Medium	The relayer list could be cold/stale long enough to cause operational failure.	Fixed through refreshed readiness and registry-backed checks.
M-CORE-07	Medium	Fee changes could be front-run against a user's intended transaction.	Fixed by binding expected fee data into the authorized transaction path.
M-CORE-08	Medium	The indexer lacked complete reorganization handling.	Fixed and later strengthened for nullifiers, checkpoints, and fork hashes.
L-CORE-01	Low	A note could be created for a recipient unable to recover/open it.	Addressed in recipient/key validation and documented compatibility boundaries.
L-CORE-02	Low	Nullifiers omit tree identity.	Not fixed; retained as Accepted Risk AR-1 below.
L-CORE-03	Low	Browser CSP, source-map, URL-scheme, host disclosure, lookup, analytics, and scaling details weakened defense in depth.	Fixed or operationally closed through the later client/relayer/release rounds.
L-CORE-04	Low	Stake-token code and external token mutability could affect registry assumptions.	Documented and checked at admission/readiness boundaries; external token behavior remains an environmental risk.

Findings register - iterative external review

The identifiers below preserve the project's remediation numbering. A single row sometimes includes closely related sub-parts that were fixed in the same change train.

ID	Sev.	Finding	Resolution / principal fix
M-01	Medium	Production deployment was not cryptographically bound to the ceremony verifier.	Fixed in `744189c`: pre-deploy transcript/zkey/source gates, post-deploy proof tests, and visible runtime hash.
I-01	Info	Receipt tests signed a different time-derived object and could pass for the wrong rejection reason.	Fixed in `744189c`: one object, one signature, reason assertion, and valid control.
M-02	Medium	Hand-written commitment ABI drifted; deposit-status checks swallowed unrelated errors.	Fixed in `eeb2dc0`: shared ABI where possible, drift tests for topic/indexed flags, fail-closed errors.

ID	Sev.	Finding	Resolution / principal fix
L-01	Low	Documented dry-run/network commands could transact or run on an unintended network.	Fixed in `eeb2dc0`: working PowerShell/environment form, explicit selected-network refusal, governance-key documentation.
I-02	Info	ABI drift test could pass without observing the required event.	Fixed by naming the required event set and validating by mutation.
M-03	Medium	Case-sensitive `.env` address comparison disabled four deployment guards.	Fixed in `c4ca170`: `getAddress()` validation/normalization and invalid-destination refusals.
M-04	Medium	A readiness warning exited successfully, so publication could proceed before governance handover.	Fixed in `c4ca170`: non-zero `launch-ready` gate with explicit invariants.
M-05	Medium	Relayer setup order made readiness impossible and the readiness check trusted too little.	Fixed in `a0aff44`: corrected order; HTTP, address, chain, bond, operator, stake, and registry checks.
M-06	Medium	The claimed no-deposits-under-hot-governance boundary was not enforced by the contract.	Fixed, then strengthened in `5bf1407`: immutable cold `depositActivator`, completed-handover check, one-way activation.
M-07	Medium	Production deploy script referenced an undeclared activator and passed the wrong constructor arity.	Fixed in `fa34f72`; all deploy paths and arity/declaration regressions aligned.
M-08	Medium	Zero-value transactions could enter the unopened pool and create commitments.	Fixed in `fa34f72`: the complete transaction entrypoint is activation-gated.
M-09	Medium	Default `npm run deploy` bypassed the hardened live path and opened a hot-governed pool.	Fixed in `f1d819c`: one live entrypoint and a behaviorally enforced local-only helper.
I-03	Info	Deployment-script test claimed general undefined-name detection but only parsed syntax and constructor arguments.	Corrected in `f1d819c`; scope and token matching now match the property.
I-04	Info	Local-chain guard regression inspected source text instead of exercising behavior.	Fixed with behavioral subprocess coverage.
L-02	Low	README/launch claims and production relayer activation support had drifted.	Fixed in the launch/self-audit train ending at `b8bf00b`.
M-10	Medium	Installed standalone relayer omitted the activation gate; readiness treated missing `poolOpen` as acceptable.	Fixed in `fb866ca`; unknown state now fails closed throughout relayer, API, app, and readiness.
M-11	Medium	Reorganization repair removed orphaned notes but retained orphaned nullifiers, hiding spendable value until restart.	Fixed in `fb866ca` with block-aware nullifiers and symmetric window repair.
L-03	Low	An already-open browser page did not observe pool activation.	Fixed in `fb866ca` with bounded polling and focus refresh until open.
L-04	Low	Indexer synchronization formula could report an impossible or misleading state.	Fixed in `1437169` with explicit chain/indexed heights.
M-12	Medium	Automatic listing could defeat staged asset admission.	Fixed in `c4d6fc1`: listing and on-chain admission are separated and bound.
M-13	Medium	Public listing requests could spend the lister's gas.	Fixed in `c4d6fc1` with signed/authorized listing boundaries and refusal modes.
M-14	Medium	A bad proving-artifact response prevented mirror fallback.	Fixed in `c4d6fc1`; failure falls through safely.
L-05	Low	Admission precheck did not simulate the exact pool call.	Fixed and strengthened in `1fbf0ca`.
M-15	Medium	Staged admission was non-idempotent, later tiers could revoke earlier tiers, refusal could be skipped, and handover could become impossible.	Fixed in `1fbf0ca` with monotonic/idempotent sequencing and enforced prerequisites.
M-16	Medium	Mirror/browser policy and build-time checks did not prove the deployed app could safely use the artifacts.	Fixed in the mirror/CSP/build gate train beginning `1fbf0ca`.
M-18	Medium	Configuring a listing key was treated as proof that it was bound to the on-chain role.	Fixed in `5dc17f6` with on-chain key binding and route-level checks.
M-19	Medium	Zero-lister policy was enforced only at the final state, leaving unsafe intermediate transitions.	Fixed in `655e116` with transition-level enforcement.
M-20	Medium	Mirror gate checked status/CORS but not content identity and was not a hard build dependency.	Fixed in `655e116` and successively strengthened through M-26/M-27.

ID	Sev.	Finding	Resolution / principal fix
M-21	Medium	Invalid `LISTER` configuration could be discovered only after deployment.	Fixed in `5dc17f6` with pre-deployment refusal.
L-06	Low	Unsimulated asset admission was not visible in the operational output.	Fixed in `5dc17f6`.
L-07	Low	Mirror probe accepted over-delivery.	Fixed in `5dc17f6` with exact response validation.
L-08	Low	Cold-wallet funding instructions covered one transaction although activation required two.	Fixed in `5dc17f6`.
L-09	Low	Mirror validation tested only the production origin, not preview/browser origins.	Fixed in `5dc17f6`, then made byte-invariant per origin in `5648705`.
L-10	Low	HTTP mirror behavior accepted by Node could still be blocked by a browser.	Fixed in `5dc17f6` with browser-relevant validation.
M-23	Medium	Listing-key binding was cached only at startup and stayed stale after rotation.	Fixed in `ab9b697`, then route-bound in `daf2428`.
M-24	Medium	CORS gate judged `HEAD`, while the browser used `GET`.	Fixed in `ab9b697`.
L-11	Low	Unsimulated admission provenance was lost on reload.	Fixed in `ab9b697`.
M-25	Medium	The real route latched a stale negative listing-key result while the recovery test bypassed that route.	Fixed in `daf2428`: duplicate guard deleted; real HTTP route tests added.
M-26	Medium	A ranged one-byte probe did not certify the browser's full artifact `GET`.	Fixed in `daf2428`: full size/hash verification for every accepted response.
L-12	Low	Importing the mirror judge executed the CLI and could set a failure exit code.	Fixed in `daf2428` with an entrypoint guard.
M-27	Medium	Wildcard CORS proved readability, not body invariance across origins.	Fixed in `5648705`: every relevant origin is fetched and hashed.
L-13	Low	Most mirror failures omitted the failing origin, and tests accidentally pinned the omission.	Fixed in `ec4c9bf`: caller-owned origin context and a sweep over all failure classes.
M-28	Medium	Readiness trusted the API's self-reported synchronization state.	Fixed in `684599f`, then made fork-bound and mandatory through M-32/M-34.
M-29	Medium	Aggregate note counts did not prove that the exact live-flow outputs had been indexed.	Fixed in `684599f` by checking exact commitments/nullifiers.
M-30	Medium	Existing accrued fees corrupted live-flow deposit delta assertions.	Fixed in `684599f` by measuring the correct balance/fee deltas.
M-31	Medium	Live-flow preflight was weaker than the publication gate.	Fixed in `684599f`; the shared validated caller path was completed in `7598104`.
L-14	Low	Cleanup instructions told operators to delete recovery keys despite residual gas/dust risk.	Fixed in `684599f` with safe retirement guidance.
L-15	Low	Issuer documentation overstated the role's powers.	Fixed in `684599f`.
M-32	Medium	API sync could accept impossible ahead/invalid positions and a height-only claim did not establish chain identity.	Fixed in `7598104`; mandatory checkpoint binding completed in `ef58ce0`.
M-33	Medium	Live-flow retained a default tolerance inconsistent with the hardened gate.	Fixed in `7598104`.
L-16	Low	The project claimed a shared relayer validator, but callers still diverged.	Fixed in `7598104` by using the actual shared caller/validator.
L-17	Low	Watchdog treated malformed HTTP 200 output as healthy.	Fixed in `7598104` with strict body/schema validation.
M-34	Medium	API readiness checkpoint was optional and not necessarily bound to the reported scan cursor; a forked or stale API could still appear ready.	Fixed in `ef58ce0`: required checkpoint, exact cursor equality, block hash match, atomic producer updates, and multi-batch/reorg/failure harness coverage. Verified again at final `e2d2c2d`.

Numbering gaps reflect the project's preserved issue labels and grouped remediation replies; they do not represent omitted open findings.

Project self-audit items reviewed

The project also disclosed and remediated defects found while checking its own fixes. These are included because they affected assurance even when not assigned an external M/L number:

- added a real `no-undef` lint rule after recognizing the narrower constructor-token test;
- repaired activation support across app/API surfaces after the gate changed their assumptions;
- executed and covered local deployment paths rather than only parsing them;
- corrected stale or inaccurate revert messages and launch claims;
- added durable recovery files for live-flow note keys and stopped discarding exit-wallet keys;
- corrected assertions that confused fee accrual with fee payment;
- repaired client note-picker depth and cumulative-count wording regressions;
- replaced comments and warnings with executable gates where the property was security-critical.

These fixes improved the evidence, but they also demonstrate why this report preserves limitations: several defects occurred in fixes to earlier defects, and green tests sometimes exercised the wrong layer or asserted too little.

Accepted risks and residual design constraints

AR-1 - nullifier not bound to tree identifier

Status: Accepted by the project; not fixed.

The nullifier is derived from the commitment, path index, and spend key but not the rotating tree identifier. If the same commitment occupies the same leaf position in different trees, spending one produces the same nullifier as the other and permanently freezes the second note. Honest accidental repetition is extremely unlikely with a correctly generated 248-bit blinding value. An adversary can choose duplicate note material, although controlling the same position across rotations is materially harder and may require timing or transaction-order influence. The consequence is denial of spendability for the colliding note, not unauthorized spending of unrelated notes.

The project accepted the risk because contract-side duplicate prevention or tree-bound nullifiers add storage or circuit/interface cost. Recommended operational posture: preserve high-quality randomness, never intentionally reuse complete note material, monitor duplicate commitments, and reconsider tree binding in any circuit/version upgrade.

AR-2 - no privileged recovery, pause, upgrade, or migration

The deployed design intentionally has no administrator path that can withdraw user value, repair a note, pause after launch, upgrade the pool, or migrate balances. This reduces custody and coercion risk but makes failures irreversible. A token that blacklists or pauses the pool, a verifier defect, lost note secrets, or an undiscovered circuit/contract bug can permanently strand value. Asset admission, code review, ceremony discipline, and user warnings are the mitigations; they are not recovery mechanisms.

AR-3 - privacy depends on usage and operational metadata

Asset type is public. Amounts and parties are hidden in the proof system, but low-volume assets, first deposits, timing, relayer/network metadata, public withdrawals, distinctive amounts at external boundaries, and client telemetry can reduce practical anonymity. One cross-asset tree enlarges the cryptographic set but does not make every participant behaviorally indistinguishable.

AR-4 - external asset and infrastructure behavior

Admission is a judgment about a token at a point in time. Upgradeable, pausable, blacklistable, rebasing, fee-charging, or malicious tokens can change later. RPC nodes, mirrors, CDNs, relayers, browsers, and hosting can fail or equivocate. The reviewed implementation generally fails closed and checks exact balances/content, but availability remains externally dependent.

Clarifications, challenged claims, and withdrawn portions

No finding was silently deleted because it was inconvenient. The following review conclusions were narrowed or challenged during the engagement:

- The deposit-side fee-on-transfer concern was narrowed: exact inbound balance checks reject a short receipt, so it does not drain other assets. The outbound/non-standard-token lock or accounting risk was real and was fixed with exact outbound checks and admission controls.
- `Num2Bits(levels)` was confirmed as sufficient to canonicalize `pathIndices` in the reviewed circuit, provided it remains connected exactly as reviewed. Removing or bypassing it would be a critical soundness regression.
- The tree-id nullifier concern was not dismissed solely as a `2^-248` event. Random accidental collision is negligible, but adversarial duplicate material changes the threat model. The project retained it as AR-1 rather than calling it fixed.
- Some artifact, readiness, and mirror statements were initially broader than the code or tests established. The final conclusion relies on the later executable gates, not those earlier statements.
- Several test claims were corrected where they parsed source, called a service directly, or asserted only a request count while the security property lived at a script, route, browser request, or orchestration boundary.

Closure evidence

The final closure review verified the repository was clean at `e2d2c2d7eb9c0cb59d3827d452d38d5feecf02f4` and reviewed the complete M-34 delta from `7598104` through `ef58ce0`, plus the final comment/documentation commits. Targeted checks established:

- missing, stale, unbound, or wrong-hash API checkpoints are rejected;
- a checkpoint is accepted only when its block equals the reported scanned block and its hash matches the chain provider;
- the indexer advances cursor and checkpoint together after obtaining the block hash;
- rewind, multi-batch, unreadable-block, stale-checkpoint, and fork-hash cases have dedicated harness coverage;
- relevant JavaScript entrypoints parse successfully;
- no new actionable security issue was identified in the final remediation.

Project-provided closure evidence reported 330 contract/script tests, 126 API tests, and 65 app tests passing; real proof construction and earlier live pool flows; and a final `launch-ready` result of 17/17. The final remediation was also checked live against a bound checkpoint on chain 4663. `live-flow.js` was not rerun end to end after the final checkpoint change because the deployment wallet held only 0.0032 ETH; its deciding predicates were extracted and covered instead. These results were reviewed as evidence but were not all independently rerun in the final closure turn.

Point-in-time deployment evidence

The following identifiers were observed or supplied during the engagement and are included for reproducibility, not as a continuing uptime guarantee:

- Pool: `0x521d73dC5d805bF5F3856f20f90e6acED1907416`
- Registry: `0x6584556aace59255c239aACb96FAe1698Aca008C`
- Relayer operator: `0xcE093c8Bb07C43A7158147FCb6830ea8ce3102F0`
- Ceremony final zkey SHA-256: `90c2676d304f60c07ea1f9b5bc2a2762ac54fe5e0bc9d1749d379112d29c87b9`
- Compiled/deployed verifier runtime SHA-256: `1337b083f02dcc99cf01c8966a39fe41a7862f7d7a3ebea38c3710951558d50d`

Addresses, code, governance, registries, relayers, hosted artifacts, and chain state can change after this report. Users should independently verify current state against the reviewed commit and published hashes.

Final assessment

Within the scope and limitations above, the final reviewed commit materially enforces the intended Nought security model across circuits, contracts, deployment, activation, relaying, indexing, listing, artifact delivery, and browser consumption. All implementation findings raised during this engagement were closed at the reviewed commit. The accepted tree-id/nullifier collision risk and the system's intentionally irreversible design remain.

The appropriate public statement is:

> Nought was reviewed by an independent Robinhood contributor in an AI-assisted security review using OpenAI Codex. The review covered the source and operational security boundaries identified in this report at commit `e2d2c2d7eb9c0cb59d3827d452d38d5feecf02f4`. This is a point-in-time review, not a warranty or formal certification. One documented nullifier/tree-rotation risk remains accepted by the project.

It would be inaccurate to say that OpenAI, Codex, or Robinhood certified, endorsed, or guaranteed the protocol.

Publication and integrity

The project may publish this report unmodified. If excerpts are used, the disclosure, final reviewed commit, limitations, and accepted risks should remain adjacent to any claim that the project was audited or reviewed.

The companion `Nought-Security-Review-SHA256.txt` contains SHA-256 hashes for the Markdown and PDF report files. Verify the hash before relying on a redistributed copy. The manifest is not itself a signature; obtain it through an independently trusted channel if tamper evidence is required.

Re-review trigger

A new review is recommended for any change to circuit constraints, proving/verifying keys, verifier generation, commitment or nullifier derivation, Merkle-tree rotation, public inputs, token accounting, governance/activation, listing/admission, indexer checkpoints, relayer receipt semantics, artifact mirrors, deployment scripts, or production addresses. Dependency or compiler upgrades should also trigger focused revalidation.