Kyte

Security Self-Assessment

Automated analysis, manual code review and on-chain configuration checks of Kyte's deployed smart contracts on Robinhood Chain.

PREPARED BY	DATE	VERSION	CHAIN
Kyte team	9 November 2026	1.1	Robinhood Chain (4663)

SNAPSHOT BLOCK
109,974,684

> **This is a self-assessment, not a third-party audit.** The Kyte team ran two open-source security analysers on the exact verified source of each deployed contract, reviewed every result by hand, and checked the live configuration on chain. Slither and Aderyn are tools by Trail of Bits and Cyfrin; neither company reviewed Kyte.

1. Summary
2. System overview
3. Scope
4. Roles and privileges
5. Live configuration
6. Methodology
7. Severity definitions
8. Findings
9. Contract review notes
10. Checks that passed
11. False positives
12. Limitations
13. Reproduce

1. Summary

Twelve deployed contracts were in scope, with about 1,750 lines of Kyte code (excluding comments). The review found **0 critical, 0 high, 0 medium, 5 low and 5 informational** issues.

The core option escrow is sound. Collateral can only go to its seller or to the option holder. Exits keep working while the protocol is paused. Every entry point is protected against reentrancy, and fees are capped in code.

The five low findings cover:

- the on-chain buyback route, which is not in use (buybacks are currently executed manually)
- a griefing case on the ETH convenience Router, which has a direct-WETH fallback
- a timelock delay that differs from the code comments
- the dependency of physical settlement on token transferability

None of them allows anyone to take escrowed collateral or staked $KYTE.

0	0	0	5	5
CRITICAL	HIGH	MEDIUM	LOW	INFO

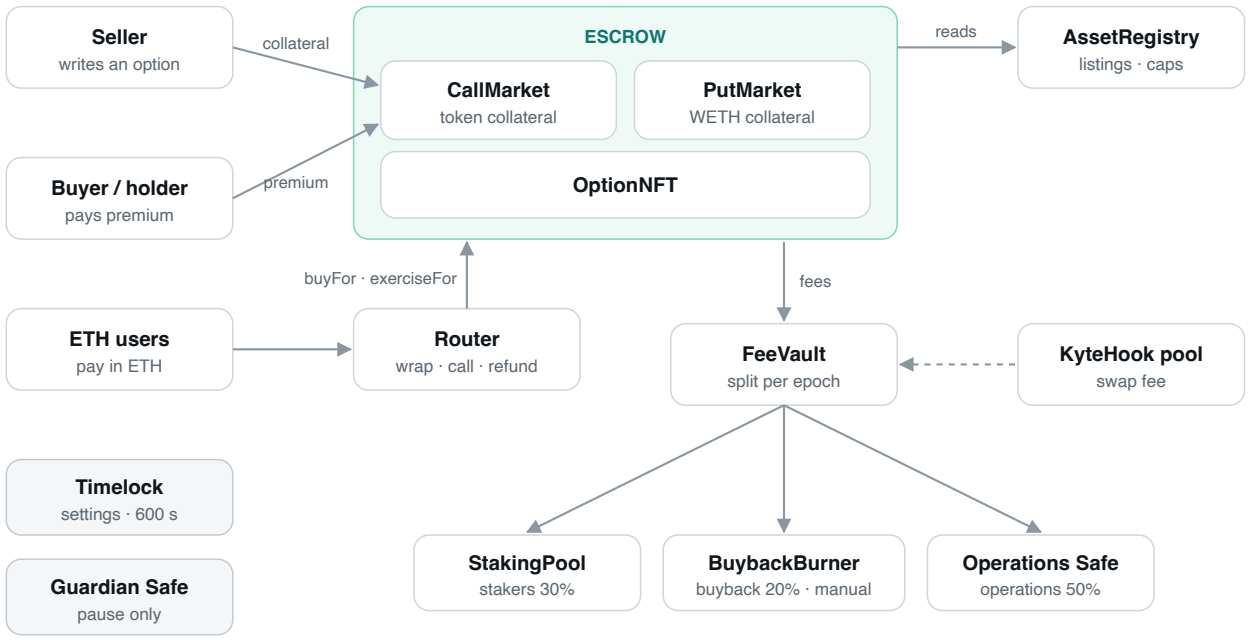
ID	SEVERITY	TITLE	STATUS
KYT-01	LOW	The on-chain buyback route points to an unfunded pool (route not in use)	Acknowledged
KYT-02	LOW	The on-chain buyback price floor follows the last fill (route not in use)	Acknowledged
KYT-03	LOW	The Router can be blocked by sending it 1 wei of WETH (direct market access unaffected)	Acknowledged
KYT-04	LOW	The timelock delay is 10 minutes on chain; the code comments describe 48 hours	Acknowledged
KYT-05	LOW	Physical settlement depends on the underlying token staying transferable	Acknowledged
KYT-06	INFORMATIONAL	The guardian Safe needs a single signature	Acknowledged
KYT-07	INFORMATIONAL	StakingPool.notifyReward emits no event	Acknowledged
KYT-08	INFORMATIONAL	KyteHaltGate reverts instead of answering if the oracle reports a future timestamp	Acknowledged
KYT-09	INFORMATIONAL	Rebasing tokens would break collateral accounting	Acknowledged
KYT-10	INFORMATIONAL	Live contracts were built from different commits	Acknowledged

2. System overview

Kyte is a fully collateralised options market:

- A seller escrows the full collateral for a covered call (the token) or a cash-secured put (WETH strike).
- A buyer pays the premium and receives an ERC-721 that represents the option.
- The holder of that NFT can exercise before expiry.
- After expiry anyone can return unexercised collateral to the seller.
- Fees flow to a vault that splits them each epoch between stakers, buyback and operations.

Settings sit behind a timelock, and a guardian can only pause new exposure.



Contract architecture and fund flows. Solid arrows are value or calls; the dashed arrow is the KyteHook protocol fee. Percentages are the live fee split. The timelock owns settings and the guardian can only pause; neither can move escrowed collateral.

3. Scope

The live Kyte contracts on Robinhood Chain at block 109,974,684. Each address resolves to a full Sourcify match, so the code analysed is the code that runs.

CONTRACT	ADDRESS	SOLC	VERIFICATION
CallMarket Covered calls: escrow, buy, exercise, expire	`0xb6759D8893601BB7c1A032eC8ea40C856a6adf80`	`0.8.28`	Full match Sourcify #58463816
PutMarket Cash-secured puts: escrow, buy, exercise, expire	`0x2E2d5DfB4cf30d15688Cc2DDD30641c78DFfcb35`	`0.8.28`	Full match Sourcify #58463839
Router ETH entry point that wraps, calls a market and refunds	`0xdD76956667E6423bAC56d2EbF827a9b0d7dBDF68`	`0.8.28`	Full match Sourcify #58463883
OptionNFT ERC-721 that represents each filled option	`0xe0cd21a7ed37898a5acdf7dc68c52259cd63308b`	`0.8.28`	Full match Sourcify #58463804
AssetRegistry Token whitelist, caps and minimum strikes	`0x02f571e10B9c7303EB7A6284A7137b146e18658D`	`0.8.28`	Full match Sourcify #58327992
FeeVault Collects fees and splits them each epoch	`0x78c569a1137b78cb78f42c42b4b6c2b8f1e3ddef`	`0.8.28`	Full match Sourcify #57936203
StakingPool $KYTE staking and WETH rewards	`0xD7253a90fd43507F44cF0934F3D5d7432760d17f`	`0.8.28`	Full match Sourcify #58609771
BuybackBurner On-chain buyback and burn (not in use; buybacks are manual)	`0x8a9d128247d8c4512411cf9d0cdcac5acf86fb95`	`0.8.28`	Full match Sourcify #58609773
UniswapV4Adapter Swap route used by BuybackBurner	`0xfa17818515e946180b9767f9d9c40a09ee80dda6`	`0.8.26`	Full match Sourcify #58609785
KyteHook Uniswap v4 hook: TWAP oracle and protocol fee	`0x77e52e93aa0892b29a4527af4f3a79f01cfa9044`	`0.8.26`	Full match Sourcify #57936350
KyteHaltGate Reader of Harmonic's session oracle (not called by live contracts)	`0x3Ba49951cae0e9EF38839F9080b48bD254e0bc8b`	`0.8.28`	Full match Sourcify #58822304

CONTRACT	ADDRESS	SOLC	VERIFICATION
TimelockController OpenZeppelin timelock that owns settings (configuration reviewed)	`0xf8e6faf7e0e57bc91ad5c59b84fede2cab842893`	`0.8.28`	Full match Sourcify #57936139

Out of scope: the $KYTE token contract (not verified on Sourcify at the time of review), the third-party pool in which $KYTE trades, the Uniswap v4 PoolManager, Harmonic's oracle, and off-chain systems (website, indexer, MCP server, keeper scripts).

4. Roles and privileges

ROLE	HOLDER	CAN DO
TimelockController `0xf8e6…2893`	Minimum delay 600 s (10 min). The timelock administers itself. The deployer holds no timelock role.	Fees for new positions within the hard caps · Router · guardian · token listings, caps and minimum strikes · FeeVault destinations and split · burner keeper, adapter and bounds · staking minimum claim · unpause
Guardian `0x9f43…0ef0`	Safe, 3 owners, threshold 1	Pause new exposure · switch a listing or its puts off. It cannot unpause, enable anything or move funds.
Operations Safe `0xd976…ada5`	Safe, 3 owners, threshold 2	Receives the operations share (50%) of each fee split. It has no protocol permissions.
Buyback keeper `0x3263…d455`	Externally owned account	Only `BuybackBurner.buyAndBurn`, within on-chain size and slippage bounds. Not used: buybacks are manual.
Router `0xdD76…DF68`	Contract set by the timelock on both markets	The `*For` functions: it pays from its own WETH, and exercise still checks NFT ownership.
FeeVault `0x78c5…ddef`	Contract	The only caller allowed to notify StakingPool of rewards.
OptionNFT deployer —	One-time	Called `setMarkets` once. It is now locked to CallMarket and PutMarket.

The timelock's proposer is the team's admin account. It is not the deployer, the operations Safe or the guardian. Public RPC logs did not identify it in this review.

5. Live configuration

CONTRACT	SETTING	VALUE AT BLOCK 109,974,684
Markets	Premium fee / exercise fee	`3% / 0.1% (caps 10% / 1%)`
Markets	Paused	`No (CallMarket, PutMarket, StakingPool)`
Markets	Router set on both markets	`0xdd76…df68`
Markets	Option ID tags (from `idOf(1)`)	`Calls 5, puts 6`
OptionNFT	Markets wired	`0xb675…df80 (calls), 0x2e2d…cb35 (puts)`
AssetRegistry	Listed tokens	`251 (maximum 300)`
FeeVault	Split	`Stakers 30% · buyback 20% · operations 50%`
StakingPool	Active stake	`13,022,690 KYTE`
Router	WETH / ETH held	`0 / 0`
BuybackBurner	Swaps executed / last price	`0 WETH in, 0 burned / 0`
BuybackBurner	Max per call / max slippage	`1 WETH / 5%`
Swap pool (KyteHook)	Liquidity / quote for 1 WETH	`0 / 997,000 KYTE`
Timelock	Minimum delay	`600 seconds`

6. Methodology

1. **Source.** Verified sources and compiler settings were downloaded from Sourcify for every address, keeping original file paths. Contracts were analysed one at a time, because live contracts come from different commits (KYT-10).

2. **Static analysis.**
    - **Slither 0.11.4** (Trail of Bits, 100 detectors) and **Aderyn 0.6.8** (Cyfrin) were run on each contract.
    - The compiler settings matched each deployment: solc 0.8.28 or 0.8.26, via-IR, 200 optimiser runs, EVM version cancun.
3. **Manual review.** Every tool result was read against the code and classified as a finding, accepted design, or a false positive. Escrow, buy, exercise, expiry, cancellation, fees, staking, buyback and governance paths were reviewed line by line (section 9).
4. **On-chain verification.** Live settings, roles, balances and pool state were read at block 109,974,684 through a public Robinhood Chain RPC (sections 4 and 5).

CONTRACT	SLITHER H	M	L	INFO	ADERYN H	L	NSLOC
CallMarket	0	0	7	17	0	9	430
PutMarket	0	0	7	17	1	9	437
Router	0	2	8	18	3	10	568
OptionNFT	2	16	0	49	0	8	362
AssetRegistry	0	0	1	3	0	6	263
FeeVault	0	1	1	17	1	4	224
StakingPool	1	9	7	30	0	6	326
BuybackBurner	0	3	3	17	1	8	243
UniswapV4Adapter + KyteHook	5	44	7	58	0	7	325
KyteHaltGate	0	5	4	1	0	2	72

Raw counts include results inside OpenZeppelin and Uniswap library code, and the same shared file is counted once per contract that imports it. Confirmed findings are in section 8 and classified false positives in section 11.

7. Severity definitions

CRITICAL	Direct loss or permanent freezing of user funds, with no special conditions.

HIGH	Loss of user funds or insolvency under realistic conditions.
MEDIUM	Loss of funds under specific conditions, or a lasting break of a core function.
LOW	Limited impact, conditions that are unlikely or inactive, or an auxiliary path with a working alternative.
INFORMATIONAL	Best practice, documentation or design notes, with no direct impact.

Severity weighs impact against likelihood. A path that is not in use is rated by its current exposure, and the rating it would carry if enabled is stated in the finding.

8. Findings

LOW **KYT-01** Acknowledged

The on-chain buyback route points to an unfunded pool (route not in use)

BuybackBurner `0x8a9d…fb95` · UniswapV4Adapter `0xfa17…dda6` · KyteHook pool `0x9689…1752`

`BuybackBurner.buyAndBurn` swaps through `UniswapV4Adapter`, which trades in the KyteHook $KYTE/WETH pool (fee 0.30%, tick spacing 60). At block 109,974,684 that pool holds zero liquidity. Its spot price and its 10-minute TWAP both read 997,000 KYTE per WETH, which is its initialisation price less the LP fee. $KYTE's live market is a different Uniswap v4 pool (`0xe25a…229e`), where 1 WETH buys about 33.8 million KYTE.

> **Context.** Kyte currently executes $KYTE buybacks manually. Platform volume has not yet justified automated buybacks, and BuybackBurner has never been called: no WETH has been swapped and `lastPrice` is 0. The rating reflects that this path is inactive. The same issue would be rated Medium if the path were switched on unchanged.

Impact if enabled. With no liquidity, a buyback call simply reverts. If liquidity were later added to the KyteHook pool near its current price, the on-chain slippage floors would accept a fill far above market. That is because the adapter quote is taken from the same pool. The keeper's `minTokenOut` would then be the only protection.

Recommendation. Before automated buybacks are switched on, point the burner at an adapter for the live pool (`setAdapter`, through the timelock), or seed the KyteHook pool at market price and let a TWAP history build. Derive `minTokenOut` from the live market.

Team response. Buybacks remain manual. The route will be corrected before automated buybacks are enabled.

LOW **KYT-02** Acknowledged

The on-chain buyback price floor follows the last fill (route not in use)

BuybackBurner `0x8a9d…fb95`, `buyAndBurn` lines 90–93 and 102

After each swap the burner records `lastPrice`, the tokens received per WETH. Every later call must satisfy `minTokenOut ≥ wethIn × lastPrice × (1 − maxSlippageBps)`. If $KYTE rises in price by more than `maxSlippageBps` (5% now, 20% at most) between two buybacks, the next call reverts. `lastPrice` cannot be reset, and WETH in the burner can only leave through a swap.

Context. Kyte currently executes $KYTE buybacks manually. Platform volume has not yet justified automated buybacks, and BuybackBurner has never been called: no WETH has been swapped and `lastPrice` is 0. The rating reflects that this path is inactive. The same issue would be rated Medium if the path were switched on unchanged.

Recommendation. Before enabling automated buybacks, bound each fill against the TWAP only, use a floor that decays over time, or add a timelock-only reset. Then deploy the revised burner and point `FeeVault.setDestinations` at it.

Team response. Acknowledged. Buybacks remain manual and will move to a revised burner before automation is enabled.

LOW **KYT-03** Acknowledged

The Router can be blocked by sending it 1 wei of WETH (direct market access unaffected)

Router `0xdD76…DF68` , `_finish` line 77

After every call the Router requires its WETH and ETH balances to be exactly zero. Anyone can transfer WETH to it, and ETH can be forced in with `SELFDESTRUCT` . The Router has no sweep function and is immutable. A 1-wei transfer would therefore make `buyWithETH` , `exerciseWithETH` , `buyPutWithETH` and `createPutOfferWithETH` revert.

Impact. This affects availability only. No funds can be taken, and positions are unaffected. The markets remain usable directly with WETH (`buy` , `exercise` , `exercisePut` , `createPutOffer`). The timelock can install a replacement Router with `setRouter` on both markets. Both of the Router's balances are 0 at block 109,974,684.

Recommendation. In the next Router, compare against balances recorded at the start of each call, or forward any surplus to the FeeVault.

Team response. Acknowledged. Direct WETH access and a timelock replacement are the fallback; the fix is planned for the next Router.

LOW **KYT-04** Acknowledged

The timelock delay is 10 minutes on chain; the code comments describe 48 hours

TimelockController `0xf8e6…2893` · `Governed.sol` line 7

`getMinDelay()` returns 600 seconds. The NatSpec in `Governed.sol` describes "the timelock (48 h, behind the admin Safe)". The timelock's powers are listed in section 4; none of them can move escrowed collateral or staked $KYTE. A short delay gives users less time to react to a settings change, such as new FeeVault destinations.

Recommendation. Either raise the delay with `updateDelay` (called through the timelock), or align the documentation with the value actually in use.

Team response. Acknowledged. The documentation will state the on-chain delay, and the delay will be reviewed as usage grows.

LOW **KYT-05** Acknowledged

Physical settlement depends on the underlying token staying transferable

CallMarket `_exercise` · PutMarket `exercisePut` · OptionMarketBase `_returnCollateral`

Exercise moves the underlying token itself: a call delivers it to the holder, and a put pulls it from the holder. Expiry and cancellation return it to the seller. If a token's issuer paused or restricted transfers, those calls would revert. A pause that overlaps an option's expiry could cost the holder the exercise window. This is inherent to physical settlement and is not a code defect.

Recommendation. Explain this in the interface next to the session warnings. A trading halt is not the same as a token transfer pause.

INFORMATIONAL **KYT-06** Acknowledged

The guardian Safe needs a single signature

Guardian `0x9f43…0ef0`

The guardian is a Safe with 3 owners and a threshold of 1. It can only pause and switch listings off, never unpause, enable anything or move funds. A single signer suits a fast circuit breaker.

INFORMATIONAL **KYT-07** Acknowledged

StakingPool.notifyReward emits no event

StakingPool `notifyReward` line 117

The reward amount is recorded by `FeeVault.FeesSplit` in the same transaction, so indexers can still follow it.

INFORMATIONAL **KYT-08** Acknowledged

KyteHaltGate reverts instead of answering if the oracle reports a future timestamp

HarmonicHaltGate `gateVerdict` line 81

If a row's `writtenAt` were ever in the future, `uint64(block.timestamp) - writtenAt` would underflow. That revert happens inside the `try` success block, so `catch` does not handle it. The behaviour fails closed. No live contract calls the gate.

INFORMATIONAL **KYT-09** Acknowledged

Rebasing tokens would break collateral accounting

OptionMarketBase `locked` / `exposure`

Fee-on-transfer tokens are rejected on chain. A rebasing token would make `locked` disagree with the real balance. Every listing goes through the timelock (251 tokens listed), and that curation is the control.

INFORMATIONAL **KYT-10** Acknowledged

Live contracts were built from different commits

Router · StakingPool · AssetRegistry

The markets use the newest constants (option ID tags 5 and 6, confirmed on chain). AssetRegistry uses an intermediate set, and Router and StakingPool were built from an earlier commit. Each contract was verified and analysed on its own source. The differences are only in constants those contracts do not use. Tagging one release commit per deployment would make future reviews simpler.

9. Contract review notes

OptionMarketBase · CallMarket · PutMarket

Offer creation validates the asset, amount, strike floor, minimum premium, duration (1 hour to 90 days), offer deadline, exposure cap and the per-seller open-offer limit, and then pulls the exact collateral. A buy sets `Filled` before minting the NFT and moving the premium. Exercise requires the NFT holder before expiry. Expiry is open to anyone at or after expiry and returns collateral to the seller. Cancellation is allowed for the seller only while the offer is open. The status always moves to its final value before any transfer, and every entry point is `nonReentrant`. Fees are fixed per position when it opens. The only places collateral can go are the seller and the option holder.

Router

The Router is stateless. It wraps exactly the ETH needed, calls the market for `msg.sender` and refunds the rest to `msg.sender`. The markets accept `*For` calls only from the router the timelock set, and `exerciseFor` still checks NFT ownership. The one issue found is KYT-03.

OptionNFT

Each ID carries a market tag in its top byte, and only that market can mint or burn it. The NFT uses `_mint`, not `_safeMint`, so a buyer cannot re-enter. `tokenURI` builds the metadata on chain and only embeds alphanumeric symbols, so a token symbol cannot inject JSON or SVG.

AssetRegistry

Only the timelock can list a token, enable it or enable puts. The guardian can only switch them off. Puts need both flags. Listing is capped at 300 tokens, and caps and minimum strikes apply per market.

FeeVault

Anyone can call `split`, at most once per epoch, and it works while paused. The split must sum to 100% and stakers always get at least 20% (live: 30% / 20% / 50%). Rounding dust stays for the next split. Destinations are set by the timelock.

StakingPool

Rewards use a reward-per-share accumulator with 1e36 precision. New stake waits until the next epoch and is then locked for one full epoch, which prevents just-in-time reward capture. Unstaking takes pending stake first, and both unstake and claim work while paused. Rewards that arrive with no active stake are carried forward. Fee-on-transfer stake tokens are rejected.

BuybackBurner · UniswapV4Adapter · KyteHook

The burner can only swap WETH for $KYTE and burn the output, falling back to the dead address if burning fails. Calls are limited to the keeper and bounded per call. KyteHook records a 128-slot tick ring buffer for its TWAP and takes its protocol fee in WETH only. This path is not in use: see KYT-01 and KYT-02.

KyteHaltGate

The gate is immutable, has no owner and holds no funds. It answers Open only for a fresh open row, and Unreadable while the oracle seat is unpaid. No live contract depends on it.

TimelockController

This is a standard OpenZeppelin TimelockController (full Sourcify match). Its configuration was reviewed on chain: the delay is 600 s, it administers itself, and the deployer holds no role. Its code was not re-analysed.

---

10. Checks that passed

✓ **No admin path to escrowed collateral**	Collateral only ever moves to its seller, or to the option holder on exercise.
✓ **Exits are never paused**	Cancel, exercise, expire, unstake, claim and split all work while paused. Pause only blocks new exposure.
✓ **Reentrancy guarded**	Every state-changing entry point uses `nonReentrant` (transient storage). State is set to its final value before any transfer.
✓ **Fees capped in code**	Premium fee ≤ 10% and exercise fee ≤ 1%, fixed per position. Live: 3% / 0.1%.
✓ **Fee-on-transfer tokens rejected**	Collateral pulls and stakes check that the balance grew by exactly the amount transferred.
✓ **NFT minting locked to the markets**	Only the market that owns an ID tag can mint or burn it. Markets are set once and wired correctly on chain.
✓ **Fee split bounded**	Shares must sum to 100% and stakers always get at least 20%.
✓ **No just-in-time staking rewards**	New stake waits one epoch and is then locked for a full epoch.
✓ **Router holds nothing**	Refunds go only to the caller. The Router's WETH and ETH balances are 0 at block 109,974,684.

✓ **Deployer has no lingering power**	The deployer holds no timelock role, and OptionNFT market wiring is locked.
✓ **Deployed code matches the source**	Every contract in scope is a full Sourcify match (creation and runtime bytecode).

11. False positives

WHERE	DETECTOR	WHY IT DOES NOT APPLY
OpenZeppelin `Math`, `Base64`; Uniswap `FullMath`, `BitMath`, `TickMath`	incorrect-exp, incorrect-shift, divide-before-multiply	Audited library code. The XOR and shift tricks are intentional.
OptionNFT `_date`	divide-before-multiply	Integer calendar arithmetic (civil-from-days); the truncation is intended.
PutMarket `exercisePut`; FeeVault `split`; BuybackBurner `buyAndBurn`	Reentrancy: state change after external call	Calls are to trusted contracts under `nonReentrant`, with the final status set first.
Router payable functions	ETH transferred without address checks	The only ETH push is the refund to `msg.sender`.
KyteHaltGate, KyteHook, FeeVault	Strict equality	These compare enums and timestamps, which is correct. The one real strict-balance check is KYT-03.
KyteHook `consult` / `observations`	weak-prng, uninitialized-state	Modulo on a time delta, not randomness. The mapping is written in `_write`.
Expiry, offer deadlines, epochs	Block timestamp used in comparisons	Time-based by design.
`setRouter`, `setKeeper`	Missing zero-address check	Zero intentionally disables these paths.
`expireMany`, `addAssets`, `gateVerdicts`	External calls inside a loop	Batches are capped (50) or timelock-only, or the function is a view.

WHERE	DETECTOR	WHY IT DOES NOT APPLY
Router constructor, adapter `settle`	Unused return value	WETH `approve` returns true or reverts. The settle amount is checked through deltas.

12. Limitations

This review cannot prove the absence of bugs. Static analysers find known patterns, and they can miss flaws in business logic, economic design, oracle assumptions and interactions between contracts. The manual review was carried out by the Kyte team, which is not independent of the code. Findings and configuration reflect the snapshot block above, and later transactions can change them. A third-party audit is recommended before significant growth in total value locked.

13. Reproduce

```
# 1. verified source and settings, per address (Robinhood Chain, 4663)
GET https://sourcify.dev/server/v2/contract/4663/<address>?fields=sources,compilation,de

# 2. Slither 0.11.4 (Trail of Bits)
slither src/<Contract>.sol \
  --solc-remaps "@openzeppelin/=node_modules/@openzeppelin/ @uniswap/=node_modules/@unis
  --solc-args "--via-ir --optimize --optimize-runs 200 --evm-version cancun" \
  --exclude-dependencies --json <Contract>.json

# 3. Aderyn 0.6.8 (Cyfrin)
aderyn . --src src -o <Contract>.md

# 4. live configuration (eth_call on a public RPC, block 109,974,684)
getMinDelay · hasRole · getThreshold · getOwners · paused · feeBps* · router · nft · cal
stakersBps/buybackBps/opsBps · lastPrice · totalBurned · quoteSpot · PoolManager.extsloa
```

The unedited Slither (JSON and text) and Aderyn (Markdown) outputs for every contract, the scope file and the on-chain snapshot are published with this report.