



TECHNICAL SECURITY ASSESSMENT

Helion Protocol

Smart Contract Audit · Liquid Staking

REPORT ID QX-TSA-2026-009

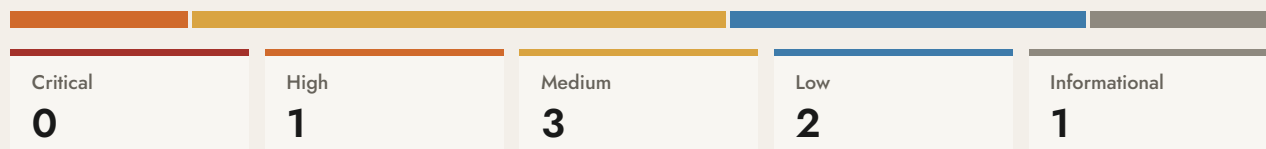
QUANTIX RESEARCH ASSESSMENT

Staking pool, rebasing hETH, operator registry, accounting oracle and withdrawal vault. 2,486 nSLOC of Solidity.

Document status	Published
Published	12 January 2026
Review period	Ecosystems
10 November 2025 – 5 December 2025	Ethereum
Languages	Review effort
Solidity	Three reviewers over twenty working days, with a parallel specification and verification track
Reviewed commit	Remediation commit
b8e4d2a07f1c935e6d0a4b28c71f9e3d5a6b0c2e	41f0c7a9e2d58b36c1e07a4d9f2b6e8c03a5d7b1

FINDINGS BY SEVERITY

7 findings · 100% remediated



This document is a Quantix research assessment of Helion Protocol, a protocol design from Quantix Protocol Research. It is not a client engagement, it does not describe a deployed system, and it is not a certification.

DOCUMENT CONTROL

Document ID	QX-TSA-2026-009
Title	Technical Security Assessment: Helion Protocol
Version	1.0
Status	Published
Assessment type	Smart Contract Audit
Review period	10 November 2025 – 5 December 2025
Published	12 January 2026
Reviewed commit	b8e4d2a07f1c935e6d0a4b28c71f9e3d5a6b0c2e

CONTENTS

1	Executive summary	3
2	Scope	4
3	Protocol architecture	4
4	Threat model	5
5	Methodology and severity classification	6
6	Findings summary	7
7	Detailed findings	8
	HP-01 Oracle quorum can fall below a majority of committee members	8
	HP-02 Batch finalisation reserves less ETH than the sum of claims, so the last claimant in a batch cannot with...	9
	HP-03 Transfers of hETH can deliver one to two wei less than the requested amount	10
	HP-04 Node operators can front-run pool deposits with their own withdrawal credentials	11
	HP-05 Withdrawal-vault ETH arriving after the reference slot is counted as rewards	12
	HP-06 Pause authority held by a single externally owned account	13
	HP-07 Quorum and membership changes are not emitted as events	13
8	Severity and risk analysis	15
9	Remediation status	15
10	Formal verification and invariants	16
11	Recommendations	17
12	Limitations	17
A	Technical appendix	18

1

Executive summary

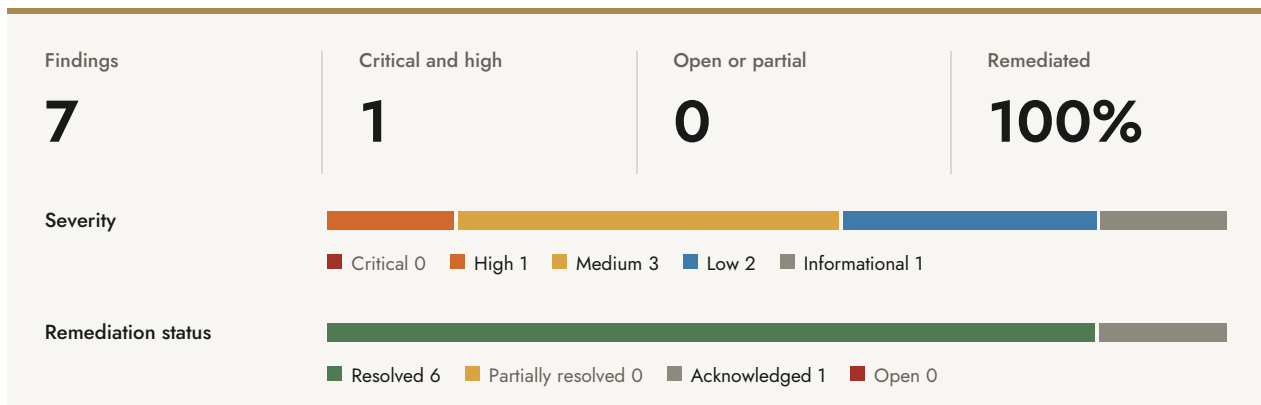
Helion Protocol lets users stake ETH without running validators. Deposits mint hETH, a rebasing token whose balances grow as validator rewards are reported. Approved node operators run the validators, and an oracle committee reports beacon-chain balances once per day. Quantix reviewed the pool, token, registry, oracle and withdrawal vault at the commit in this report.

The single high-severity finding concerned the oracle committee: the quorum could be set, or left by member removals, below a majority of members, letting a minority finalise balance reports that drive every rebase. Medium-severity findings covered rounding in withdrawal-batch finalisation that could leave the last claimant unable to withdraw, deposit front-running by node operators, and a timing mismatch between oracle reference slots and the vault balance.

All findings except a documented one-to-two wei rounding behaviour were fixed and verified. A formal specification of share accounting and quorum rules accompanies this report; all specified properties hold on the remediation commit.

Helion's main residual risks are those inherent to liquid staking: correlated slashing across operators, oracle committee honesty, and the time withdrawals take on the consensus layer.

AT A GLANCE



SEVERITY SUMMARY

Severity	Definition	Count	Open
■ Critical	Direct loss of funds or permanent loss of control is achievable with little or no precondition.	0	0
■ High	Material loss or protocol disruption is achievable under realistic conditions.	1	0
■ Medium	Limited loss, degraded functionality, or impact that depends on specific conditions.	3	0
■ Low	Minor impact, defence-in-depth gaps, or issues with unlikely preconditions.	2	0
■ Informational	Code quality, documentation or best-practice observations without direct security impact.	1	0

2

Scope

Staking pool, rebasing hETH, operator registry, accounting oracle and withdrawal vault. 2,486 nSLOC of Solidity.

Repository	github.com/example-org/helion-core
Reviewed commit	b8e4d2a07f1c935e6d0a4b28c71f9e3d5a6b0c2e
Remediation commit	41f0c7a9e2d58b36c1e07a4d9f2b6e8c03a5d7b1
Review period	10 November 2025 – 5 December 2025
Review effort	Three reviewers over twenty working days, with a parallel specification and verification track

Path	Description	nSLOC
contracts/StakingPool.sol	Deposits, share accounting, rebases and the withdrawal queue	934
contracts/token/hETH.sol	Rebasing liquid staking token over pool shares	312
contracts/OperatorRegistry.sol	Node operators, signing keys and deposit limits	486
contracts/oracle/AccountingOracle.sol	Committee reports of validator balances	402
contracts/WithdrawalVault.sol	Receives beacon-chain withdrawals	88
contracts/DepositGuard.sol	Deposit-root checks before staking (added during remediation)	264
Total		2,486

Out of scope

- Validator client software and node-operator infrastructure
- Off-chain oracle daemons that compute reports
- The Ethereum deposit contract and consensus layer
- Governance token and DAO voting contracts

3

Protocol architecture

`StakingPool` holds buffered ETH, mints shares to depositors and batches 32 ETH deposits to validators registered by operators in `OperatorRegistry`. hETH balances are computed as $\text{shares} \times \text{totalPooledEther} / \text{totalShares}$, so a rebase changes `totalPooledEther` without touching individual records.

Every 24 hours, members of the `AccountingOracle` committee submit a report for a reference slot: total validator balance, number of exited validators, and the withdrawal-vault balance. When a quorum of identical reports arrives, the pool applies it, bounded by sanity checks on the implied annual rate.

Withdrawals are requested in hETH, queued as NFTs, and finalised in batches when the oracle report shows enough

ETH has arrived in the `WithdrawalVault` through EIP-4895 withdrawals.

Component	Role	Trust level
StakingPool	Share accounting, deposits, withdrawal queue	Trusted
hETH	Rebasing token interface over shares	Trusted
OperatorRegistry	Operator keys and limits	Trusted
AccountingOracle	Applies committee reports	Trusted
Oracle committee members	Report validator balances	Semi-trusted
Node operators	Run validators with registered keys	Semi-trusted
Stakers and integrators	Deposit, transfer and withdraw	Untrusted

Trust assumptions

- A majority of oracle committee members report honestly.
- Node operators do not collude to slash validators deliberately; misbehaviour is bounded by deposit limits.
- The consensus layer processes withdrawals per its specification.
- Governance changes pass through a timelock and are monitored.

Threat model

Assets

- Staked ETH and buffered deposits
- Correct hETH supply and balances
- Withdrawal queue solvency
- Validator withdrawal credentials

Actors

Actor	Capability	Trust
Colluding oracle minority	Submits false balance reports	Semi-trusted
Malicious node operator	Front-runs deposits with its own withdrawal credentials	Semi-trusted
Rebase arbitrageur	Times deposits and withdrawals around reports	Untrusted
Committee manager role	Adds and removes members, sets quorum	Trusted

Threats

Threat	Actor and vector	Mitigation	Findings
Inflate or deflate hETH supply through a false report	Colluding oracle minority: Quorum below a majority	Quorum floor of a strict majority on every membership change; rebase bounds	HP-01

Threat	Actor and vector	Mitigation	Findings
Leave late claimants unable to withdraw	Rounding: Batch reservation rounding differs from claim rounding	Store per-request amounts at finalisation	HP-02
Steal a 32 ETH deposit	Malicious node operator: Pre-deposit with own withdrawal credentials	Guardian-signed deposit root check	HP-04
Extract value around rebases	Rebase arbitrageur: Reward double counting from vault timing	Report vault balance at the reference slot	HP-05

5

Methodology and severity classification

1. Specification workshop covering share accounting, report lifecycle and the withdrawal queue
2. Manual review of all in-scope contracts by two reviewers each
3. Invariant fuzzing of share accounting under rebases, slashing and batched withdrawals
4. Formal specification of accounting and quorum rules checked with an SMT-based prover
5. Scenario analysis of oracle collusion, operator front-running and rebase timing
6. Remediation review including the new DepositGuard contract

Severity classification

Each finding is rated on the impact it would have if exploited and the likelihood that its preconditions occur. The rating reflects the code at the reviewed commit.

Severity	Meaning
■ Critical	Direct loss of funds or permanent loss of control is achievable with little or no precondition.
■ High	Material loss or protocol disruption is achievable under realistic conditions.
■ Medium	Limited loss, degraded functionality, or impact that depends on specific conditions.
■ Low	Minor impact, defence-in-depth gaps, or issues with unlikely preconditions.
■ Informational	Code quality, documentation or best-practice observations without direct security impact.

Remediation status

Status	Meaning
Resolved	Fixed and verified in the remediation review.
Partially resolved	Mitigated; the residual risk is documented.
Acknowledged	Accepted by the protocol team with a stated rationale.
Open	Not yet addressed at the reviewed remediation commit.

6

Findings summary

ID	Title	Severity	Category	Status
HP-01	Oracle quorum can fall below a majority of committee members	High	Governance manipulation	Resolved
HP-02	Batch finalisation reserves less ETH than the sum of claims, so the last claimant in a batch cannot withdraw	Medium	Precision and rounding	Resolved
HP-03	Transfers of hETH can deliver one to two wei less than the requested amount	Low	Precision and rounding	Acknowledged
HP-04	Node operators can front-run pool deposits with their own withdrawal credentials	Medium	Business logic	Resolved
HP-05	Withdrawal-vault ETH arriving after the reference slot is counted as rewards	Medium	Business logic	Resolved
HP-06	Pause authority held by a single externally owned account	Low	Key management	Resolved
HP-07	Quorum and membership changes are not emitted as events	Informational	Code quality	Resolved

7

Detailed findings

HP-01 ■ High Resolved

Oracle quorum can fall below a majority of committee members

contracts/oracle/AccountingOracle.sol:L94-L131 (setQuorum, removeMember, _submitReport)

Category	CWE	Likelihood	Impact
Governance manipulation	CWE-285	Medium	High

DESCRIPTION

Each report is finalised when `quorum` members have submitted the same report hash for a frame. `setQuorum(q)` only requires `q > 0`, and `removeMember()` decrements the member count without re-checking the quorum. With seven members and a quorum of four, removing three members leaves four members and a quorum of four, which is fine; but a later `setQuorum(2)` or an initial misconfiguration lets two members finalise reports alone.

Reports drive `totalPooledEther`, so a finalised false report changes every hETH balance. The pool applies sanity bounds to the implied annual rate, but the bounds were generous (an annualised 27% positive rebase) and a negative report is bounded only by the slashing limit.

The defect is that the safety of the oracle depends on an invariant (quorum is a strict majority) that the contract does not enforce.

PRECONDITIONS

- A quorum set below a strict majority, through configuration error or a compromised manager role.
- Collusion or compromise of that many committee members.

EXPLOIT SCENARIO

1. The committee manager role (a 3-of-5 multisig) lowers the quorum to two during a member rotation and forgets to raise it.
2. Two colluding members submit a report overstating validator balances within the 27% annualised bound.
3. The pool applies a positive rebase. The members, who hold hETH, request withdrawals at the inflated rate before the next honest report corrects it.

IMPACT

Value transferred from stakers to the colluding members, bounded by the rebase limit per report; repeated reports compound the loss.

RECOMMENDATION

Enforce `quorum >= members / 2 + 1` in `setQuorum`, `addMember` and `removeMember`, and tighten the positive rebase bound to a realistic annualised rate.

FIX

contracts/oracle/AccountingOracle.sol (remediation commit)	DIFF
1 function setQuorum(uint256 q) external onlyRole(MANAGE_MEMBERS_ROLE) {	
2 - require(q > 0, "quorum");	
3 + _requireMajority(q, members.length);	
4 quorum = q;	
5 }	
6	
7 function removeMember(address m) external onlyRole(MANAGE_MEMBERS_ROLE) {	
8 _remove(m);	

```
9 +     if (quorum > members.length) quorum = members.length;
10 +     _requireMajority(quorum, members.length);
11 + }
12 +
13 + function _requireMajority(uint256 q, uint256 n) internal pure {
14 +     if (q < n / 2 + 1 || q > n) revert QuorumNotMajority(q, n);
15 + }
```

RESOLUTION

Fixed. The quorum is validated on every membership change, and the positive rebase bound was lowered to an annualised 10%.

REFERENCES

- CWE-285: Improper Authorization
<https://cwe.mitre.org/data/definitions/285.html>

HP-02 ■ Medium Resolved

Batch finalisation reserves less ETH than the sum of claims, so the last claimant in a batch cannot withdraw

contracts/StakingPool.sol:L611-L668 (`_finalizeBatch`, `claimWithdrawal`)

Category	CWE	Likelihood	Impact
Precision and rounding	CWE-1339	Medium	Medium

DESCRIPTION

When a batch of withdrawal requests is finalised, the pool reserves `floor(sumShares × rate / PRECISION)` ETH for the whole batch, where `rate` is the share rate applied to the batch. Each claim later computes its own amount with `Math.mulDiv(requestShares, rate, PRECISION, Math.Rounding.Ceil)`; the developers rounded claims up so that no user would receive less than their shares are worth.

The two calculations round in opposite directions. Each claim can exceed its exact share of the reservation by up to one wei, while the reservation itself is rounded down, so the sum of claims in a batch can exceed the reserved ETH by up to the number of requests in the batch. Claims are paid first come, first served out of `LockedEther`.

The shortfall is tiny, but its effect is binary: the last claimant in the batch finds the reservation exhausted and the transaction reverts. The user is blocked until someone donates the missing wei or a later batch happens to over-reserve.

PRECONDITIONS

- A batch in which at least one request's conversion is not exact, which is the normal case.

EXPLOIT SCENARIO

1. A batch of 400 requests is finalised. Rounding each claim up makes their sum 3 wei larger than the rounded-down reservation.
2. The first 399 claimants withdraw normally.
3. The final claim reverts with "insufficient reserved ether" and remains blocked.

IMPACT

Denial of withdrawal for affected users until manual intervention. Invariant P4 (queue solvency) is violated.

RECOMMENDATION

Compute each request's claimable ETH once at finalisation with a single rounding direction (down), reserve exactly the sum of those stored amounts, and pay claims from the stored value.

FIX

	contracts/StakingPool.sol (remediation commit)	DIFF
1	-	uint256 reserve = sumShares * rate / PRECISION;
2	+	uint256 reserve;
3	+	for (uint256 id = first; id <= last; ++id) {
4	+	Request storage r = requests[id];
5	+	uint256 amount = Math.mulDiv(r.shares, rate, PRECISION); // floor, stored once
6	+	r.claimable = uint128(amount);
7	+	reserve += amount;
8	+	}
9		lockedEther += reserve;
10	...	
11	-	uint256 amount = Math.mulDiv(r.shares, batchRate[r.batch], PRECISION, Math.Rounding.Ceil)
		;
12	+	uint256 amount = r.claimable;

RESOLUTION

Fixed. Claims are paid from stored per-request amounts and invariant P4 holds on the remediation commit.

REFERENCES

- CWE-1339: Insufficient Precision or Accuracy of a Real Number
<https://cwe.mitre.org/data/definitions/1339.html>

HP-03 ■ Low Acknowledged

Transfers of hETH can deliver one to two wei less than the requested amount

contracts/token/hETH.sol:L144-L172 (transfer, transferFrom, getSharesByPooledEth)

Category	CWE	Likelihood	Impact
Precision and rounding	CWE-682	Medium	Low

DESCRIPTION

hETH balances are derived from shares. A transfer of `amount` converts to shares with `amount × totalShares / totalPooledEther`, rounding down, so the recipient's balance increases by slightly less than `amount` and the sender may retain one or two wei after "transferring everything".

This is inherent to rebasing tokens built over shares. It does not lose value for the protocol, but integrations that assume `balanceOf(recipient)` increases by exactly `amount` (vaults, bridges, accounting that compares balances) can fail or record dust mismatches.

PRECONDITIONS

- An integration that relies on exact transfer amounts.

EXPLOIT SCENARIO

1. A vault records a deposit of 10 hETH but receives 9.9999999999999999 hETH; its strict accounting check reverts.

IMPACT

Integration failures and dust balances; no loss to stakers.

RECOMMENDATION

Document the behaviour prominently, provide `transferShares()` for integrators, and recommend a non-rebasing wrapper for DeFi use.

RESOLUTION

Acknowledged. Documented in the integration guide; `transferShares()` and a wrapped non-rebasing token are available.

REFERENCES

- EIP-20: Token Standard
<https://eips.ethereum.org/EIPS/eip-20>
- CWE-682: Incorrect Calculation
<https://cwe.mitre.org/data/definitions/682.html>

HP-04 ■ Medium Resolved

Node operators can front-run pool deposits with their own withdrawal credentials

contracts/StakingPool.sol:L402-L451 (depositBufferedEther), contracts/OperatorRegistry.sol:L210

Category	CWE	Likelihood	Impact
Business logic	CWE-362	Low	High

DESCRIPTION

The deposit contract keys validators by public key. If a deposit for a given public key already exists, subsequent deposits for the same key top up that validator and keep the withdrawal credentials of the first deposit.

An operator registers a signing key with Helion, observes the pool's 32 ETH deposit for that key in the mempool, and front-runs it with a 1 ETH deposit for the same key using withdrawal credentials it controls. The pool's deposit then tops up a validator whose withdrawals go to the operator.

Operators are permissioned, which lowers likelihood, but each successful front-run diverts a full 32 ETH deposit.

PRECONDITIONS

- A malicious or compromised node operator with registered keys.

EXPLOIT SCENARIO

1. The operator sees `depositBufferedEther` pending with its key.
2. It submits a 1 ETH deposit for the same key with its own withdrawal credentials and a higher priority fee.
3. The pool's 32 ETH lands on the operator-controlled validator.

IMPACT

Loss of 32 ETH per front-run deposit.

RECOMMENDATION

Before each deposit, check the deposit contract's `get_deposit_root()` against a root signed by a quorum of independent guardians, so that any deposit in between invalidates the batch.

FIX

contracts/StakingPool.sol (remediation commit)	DIFF
1 - <code>function depositBufferedEther(uint256 count) external onlyRole(DEPOSITOR_ROLE) {</code>	
2 + <code>function depositBufferedEther(uint256 count, bytes32 depositRoot, GuardianSig[] calldata sigs</code>	
3 + <code>) external {</code>	
3 + <code>if (DEPOSIT_CONTRACT.get_deposit_root() != depositRoot) revert DepositRootChanged();</code>	

```
4 + depositGuard.verify(depositRoot, count, block.number, sigs); // quorum of guardians
```

RESOLUTION

Fixed with a new `DepositGuard` contract requiring signatures from four of six guardians over the current deposit root.

REFERENCES

- Ethereum consensus specifications
<https://github.com/ethereum/consensus-specs>
- CWE-362: Race Condition
<https://cwe.mitre.org/data/definitions/362.html>

HP-05 ■ Medium Resolved

Withdrawal-vault ETH arriving after the reference slot is counted as rewards

contracts/StakingPool.sol:L702-L748 (handleOracleReport), contracts/WithdrawalVault.sol:L31

Category	CWE	Likelihood	Impact
Business logic	CWE-841	High	Low

DESCRIPTION

Reports describe the validator balance at a reference slot. When applying a report, the pool pulls the entire current balance of the `WithdrawalVault`, which also includes ETH from withdrawals processed between the reference slot and the moment the report lands on-chain (often several hours).

The validator balance in the report already excludes those withdrawals only if they happened before the reference slot. ETH that arrived later is therefore counted twice in the next report cycle and once as an unexplained reward now, producing an inflated rebase followed by a smaller one. Arbitrageurs who track withdrawals can deposit before an inflated report and request withdrawal after it.

PRECONDITIONS

- Beacon-chain withdrawals arriving between the reference slot and report submission.

EXPLOIT SCENARIO

1. An observer sees a large batch of exits processed after the reference slot, deposits before the report, and requests withdrawal at the inflated rate.

IMPACT

Small, repeated transfers of value from long-term stakers to rebase arbitrageurs.

RECOMMENDATION

Include the vault balance at the reference slot in the report and transfer only that amount when applying it.

FIX

	contracts/StakingPool.sol (remediation commit)	DIFF
1 -	<code>uint256 withdrawn = address(withdrawalVault).balance;</code>	
2 -	<code>withdrawalVault.withdrawWithdrawals(withdrawn);</code>	
3 +	<code>uint256 withdrawn = report.withdrawalVaultBalance; // at reference slot</code>	
4 +	<code>withdrawalVault.withdrawWithdrawals(withdrawn);</code>	

RESOLUTION

Fixed. Reports now carry the vault balance at the reference slot.

REFERENCES

- EIP-4895: Beacon chain push withdrawals as operations
<https://eips.ethereum.org/EIPS/eip-4895>

HP-06 ■ Low Resolved

Pause authority held by a single externally owned account

Deployment configuration: PAUSE_ROLE on StakingPool and hETH

Category	CWE	Likelihood	Impact
Key management	CWE-284	Low	Medium

DESCRIPTION

Pausing stops deposits, transfers and withdrawals. The role is held by one EOA used by the on-call engineer. Fast pausing is valuable in an incident, but a single key can also freeze user funds indefinitely if compromised, and unpausing requires the same key.

PRECONDITIONS

- Compromise or loss of the pause key.

EXPLOIT SCENARIO

1. An attacker with the key pauses hETH transfers during market stress, blocking exits.

IMPACT

Temporary freezing of user funds; no direct theft.

RECOMMENDATION

Move pausing to a guardian multisig with a low threshold and limit pause duration, with unpausing through governance.

RESOLUTION

Fixed. Pausing moved to a 2-of-5 guardian multisig; pauses expire after 14 days unless renewed by governance.

REFERENCES

- CWE-284: Improper Access Control
<https://cwe.mitre.org/data/definitions/284.html>
- OpenZeppelin Contracts: Utilities (Pausable)
<https://docs.openzeppelin.com/contracts/5.x/api/utils>

HP-07 ■ Informational Resolved

Quorum and membership changes are not emitted as events

contracts/oracle/AccountingOracle.sol

Category	CWE	Likelihood	Impact
Code quality	—	Low	Low

DESCRIPTION

Membership and quorum changes update storage without events, so monitoring cannot detect the configuration change behind HP-01 without polling storage.

PRECONDITIONS

- None; observability only.

EXPLOIT SCENARIO

1. Not exploitable.

IMPACT

Reduced ability to detect governance changes to the oracle.

RECOMMENDATION

Emit `MemberAdded`, `MemberRemoved` and `QuorumChanged`.

RESOLUTION

Fixed.

REFERENCES

- Solidity documentation: Events
<https://docs.soliditylang.org/en/latest/contracts.html#events>

8

Severity and risk analysis

The matrix places each finding by the likelihood of its preconditions and the impact of exploitation. Identifiers in each cell refer to the detailed findings in section 7.

Impact

	Low	Medium	High
High	1 HP-04	1 HP-01	0
Medium	1 HP-06	1 HP-02	0
Low	1 HP-07	1 HP-03	1 HP-05
	Low	Medium	High

Likelihood

Distribution by vulnerability class

Business logic	2
Precision and rounding	2
Code quality	1
Governance manipulation	1
Key management	1

Severity distribution



9

Remediation status

6 of 7 findings are resolved, 1 acknowledged, 0 partially resolved and 0 open, as reviewed at remediation commit 41f0c7a9e2d5.

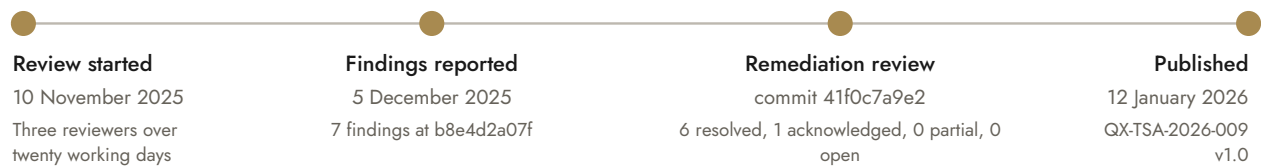
Severity	Resolved	Partially resolved	Acknowledged	Open	Total
Critical	0	0	0	0	0
High	1	0	0	0	1

Severity	Resolved	Partially resolved	Acknowledged	Open	Total
■ Medium	3	0	0	0	3
■ Low	1	0	1	0	2
■ Informational	1	0	0	0	1

Remediation progress

100%

Remediation history



Residual items

ID	Status	Notes
HP-03	Acknowledged	Acknowledged. Documented in the integration guide; <code>transferShares()</code> and a wrapped non-rebasing token are available.

10

Formal verification and invariants

Share accounting, oracle quorum and withdrawal-queue solvency were specified as rules and checked on the remediation commit. Two rules failed on the original commit and led to HP-01 and HP-02. All rules hold after remediation; a liveness property about report timing was out of reach of the prover and is covered by tests.

Tool CVL-style specification checked with an SMT-based prover; Halmos for arithmetic lemmas

ID	Property	Result	Notes
P1	The sum of shares across all holders equals <code>totalShares</code> after any operation.	■ Proved	—
P2	A report is applied only after a strict majority of current committee members submitted the same report hash.	■ Proved	Violated on the original commit (HP-01).
P3	A single report changes <code>totalPooledEther</code> by at most the configured annualised positive bound or the slashing-bounded negative limit.	■ Proved	—
P4	ETH reserved for finalised withdrawal requests is at least the sum of their unclaimed amounts.	■ Proved	Violated on the original commit by up to <code>n wei</code> per batch (HP-02).

ID	Property	Result	Notes
P5	A holder's balance changes only through transfers, rebases or withdrawals initiated by the holder.	■ Proved	—
P6	Reports are applied in order of reference slot and at most once per frame.	■ Inconclusive	Temporal property; covered by integration tests.

11

Recommendations

Monitor oracle configuration and reports

Alert on quorum and membership changes, on reports near the rebase bounds, and on disagreeing member submissions.

Diversify node operators and limit concentration

Keep per-operator deposit limits low enough that a single operator's slashing or misbehaviour is absorbed by the buffer.

Publish integration guidance for rebasing behaviour

Make the wrapped token the default for DeFi integrations and document HP-03 for those that integrate hETH directly.

Keep the formal specification in CI

Re-run the accounting and quorum rules on every change to the pool and oracle.

12

Limitations

- Off-chain oracle daemons, validator clients and operator infrastructure were out of scope.
- Consensus-layer behaviour was assumed to follow the published specifications.
- Formal results hold for the stated rules and the modelled environment; external calls to the deposit contract were summarised.

No assessment can show that a system is free of vulnerabilities. Conclusions apply to the code at the commits listed and the conditions stated in this document.

Appendix

Technical appendix

A.1 Tools

Tool	Purpose
Foundry	Unit, fork and invariant tests
SMT-based prover (CVL-style rules)	Accounting, quorum and queue-solvency properties
Halmos	Arithmetic lemmas for share conversion
Slither	Static analysis

A.2 Reviewed files (SHA-256)

File	SHA-256
contracts/StakingPool.sol	0095bdf38d1e5cb0d812578ff9efe7ea180539eb49b23b6f17d207cdc52fbc23
contracts/token/hETH.sol	5db79da03827850d55ec1f16ef11562f55ad04e2085b5dd3645fae82c6a7d9c5
contracts/OperatorRegistry.sol	3f463807e94d47ef7b6cce00c24982cbb94f1711b5f6073a75b44fe5f143b221
contracts/oracle/AccountingOracle.sol	7b3320ed8a642fbb838ec7fac9c52334fd98c4b5899b5c7ea9105b10a08b7eb6
contracts/WithdrawalVault.sol	f68726c63886a5888bd73ee5eb800006d879ea8cad49552c255a7337a824ff6a

A.3 Notes

- File hashes are SHA-256 digests of the reviewed sources at the audited commit.

A.4 Methodology and technical references

Standards, specifications and weakness classifications cited in the findings. Each entry links to the official source.

- CWE-285: Improper Authorization
<https://cwe.mitre.org/data/definitions/285.html>
- CWE-1339: Insufficient Precision or Accuracy of a Real Number
<https://cwe.mitre.org/data/definitions/1339.html>
- EIP-20: Token Standard
<https://eips.ethereum.org/EIPS/eip-20>
- CWE-682: Incorrect Calculation
<https://cwe.mitre.org/data/definitions/682.html>
- Ethereum consensus specifications
<https://github.com/ethereum/consensus-specs>
- CWE-362: Race Condition
<https://cwe.mitre.org/data/definitions/362.html>
- EIP-4895: Beacon chain push withdrawals as operations
<https://eips.ethereum.org/EIPS/eip-4895>
- CWE-284: Improper Access Control
<https://cwe.mitre.org/data/definitions/284.html>
- OpenZeppelin Contracts: Utilities (Pausable)
<https://docs.openzeppelin.com/contracts/5.x/api/utils>

10. Solidity documentation: Events

<https://docs.soliditylang.org/en/latest/contracts.html#events>

A.5 Document information

Document	Quantix Technical Security Assessment QX-TSA-2026-009, version 1.0
Subject	Helion Protocol, Quantix Protocol Research
Status	Published 12 January 2026
Disclosure	This document is a Quantix research assessment of Helion Protocol, a protocol design from Quantix Protocol Research. It is not a client engagement, it does not describe a deployed system, and it is not a certification.