Quantstamp Security Assessment Certificate

QUANTSTAMP VERIFIED SECURITY CERTIFICATE

December 14th 2020 — Quantstamp Verified

Lendroid Whalestreet

This smart contract audit was prepared by Quantstamp, the protocol for securing smart contracts.

Executive Summary

Type	Liquidity Mining
Auditors	Fayçal Lalidji, Security Auditor Leonardo Passos, Senior Research Engineer Jake Goh Si Yuan, Senior Security Researcher
Timeline	2020-11-23 through 2020-11-25
EVM	Muir Glacier
Languages	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	WhaleStreet☺ ☺Documentation
Documentation Quality	Medium
Test Quality	Medium

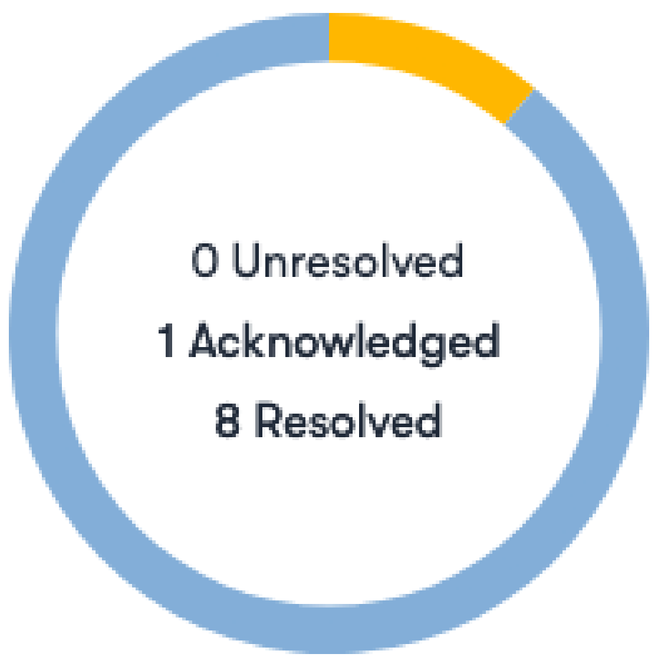
Source Code

Repository	Commit
Whalestreet-contracts	f9d5f0f

Goals

- Find issues that could lead to fund losses
- Find issues that could allow attacking staking pools
- Find inconsistencies between specification and code

Total Issues	9 (8 Resolved)
High Risk Issues	3 (3 Resolved)
Medium Risk Issues	4 (3 Resolved)
Low Risk Issues	2 (2 Resolved)
Informational Risk Issues	0 (0 Resolved)
Undetermined Risk Issues	0 (0 Resolved)



High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or is reasonably likely to lead to moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low-impact in view of the client's business circumstances.
Informational	The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
? Undetermined	The impact of the issue is uncertain.

Unresolved	Acknowledged the existence of the risk, and decided to accept it without engaging in special efforts to control it.
Acknowledged	The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).
Resolved	Adjusted program implementation, requirements or constraints to eliminate the risk.
Mitigated	Implemented actions to minimize the impact or likelihood of the risk.

Summary of Findings

Through reviewing the code, we found **9 potential issues** of various levels of severity: 3 high, 5 medium and 1 low. We recommend addressing the findings prior to deploying the smart contracts to mainnet. Quantstamp only audited the given code; all external components that the code will interface with and/or integrate with is not in the scope of this audit.
The report has been updated according to the code commit `e32d1306`. We have marked 8 of the 9 issues as resolved. During the re-audit process we discovered a new medium severity issue (QSP-9) that should be fixed or documented before the deployment.

ID	Description	Severity	Status
QSP-1	Incorrect Staking Logic	High	Fixed
QSP-2	Gas Intensive Loop Usage	High	Fixed
QSP-3	Incorrect Series 2 Reward	High	Fixed
QSP-4	Total Reward Adjustment	Medium	Fixed
QSP-5	Undistributed Epoch Zero Reward	Medium	Fixed
QSP-6	`starttime` Value Is Inconsistent With Specification	Medium	Fixed
QSP-7	Staking Manipulation	Low	Fixed
QSP-8	Input Validation	Low	Fixed
QSP-9	Incorrect Reward Per Stake	Medium	Acknowledged

Quantstamp Audit Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

The Quantstamp auditing process follows a routine series of steps:

1. Code review that includes the following
   i. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
   i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
   ii. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Toolset

The notes below outline the setup and steps performed in the process of this audit.

Setup

Tool Setup:

- Slither v0.6.14

Steps taken to run the tools:

1. Installed the Slither tool: `pip install slither-analyzer`
2. Run Slither from the project directory: `slither .`

Findings

QSP-1 Incorrect Staking Logic

Severity: High Risk

Status: Fixed

File(s) affected: `BasePool.sol`

Description: The users staked amount at epoch X will not allow them any reward at epoch Y if X < Y , meaning that their staked LP tokens are not accumulated when computing the rewards for each epoch.

Recommendation: We recommend using the users cumulative amounts and the total cumulative amounts when computing the reward for each epoch.

QSP-2 Gas Intensive Loop Usage

Severity: High Risk

Status: Fixed

File(s) affected: `BasePool.sol`

Description: If a user stakes at epoch `X`, and claims his reward at epoch `Y` using `BasePool.unstakeAndClaim` or `BasePool.claim`, and if `Y` is higher enough than `X`, transactions will most likely throw due to running out of gas or will hit the block gas limit. As a result, users will lose all their rewards.
This issue is due to the use of a loop inside `BasePool.earned` function that iterates over all previous epochs from `lastEpochRewardsClaimed` until the current epoch relative to the block timestamp.
Note that `lastEpochRewardsClaimed's` default value is zero; hence, late stakers will face the same issue, since the iteration necessarily starts from epoch zero.

Recommendation: We strongly recommend using a staking algorithm that does not use any logic based on loop.

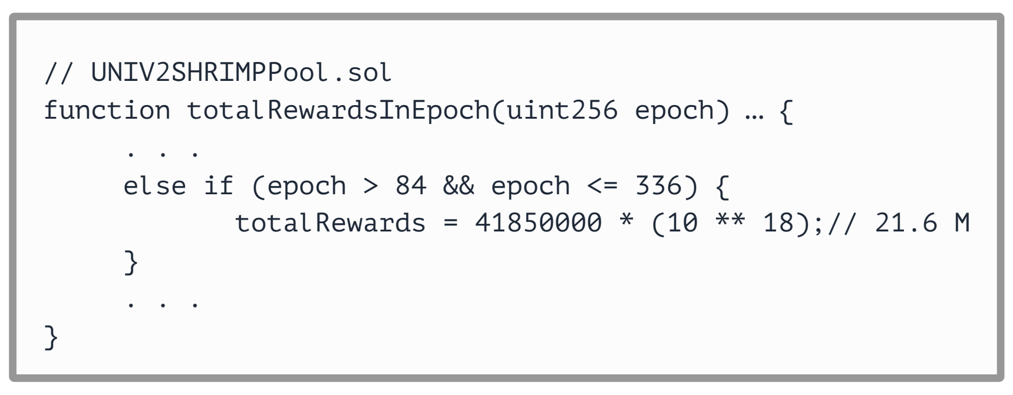
QSP-3 Incorrect Series 2 Reward

Severity: High Risk

Status: Fixed

File(s) affected: `UNIV2SHRIMPPool.sol`

Description: Total rewards between epochs 85 and 336 is almost doubled of what it should be. According to the specification and documentation in the code, the total rewards between epochs 85 and 336 should be 21.6M, but it is currently set to 41,850,000:



```
// UNIV2SHRIMPPool.sol
function totalRewardsInEpoch(uint256 epoch) … {
     . . .
     else if (epoch > 84 && epoch <= 336) {
            totalRewards = 41850000 * (10 ** 18);// 21.6 M
     }
     . . .
}
```

Recommendation: Review the rewards for epochs 85 and 336; if it is correct in the code, fix it in the specification and code comments. Otherwise, fix the code accordingly, updating comments and spec.

QSP-4 Total Reward Adjustment

Severity: Medium Risk

Status: Fixed

File(s) affected: `UNIV2SHRIMPPool.sol`

Description: In `totalRewardsInEpoch`, L88-92 adjusts `totalRewards` using a logic that is not currently documented and therefore could not be validated. As such, its effect is currently undetermined. As a consequence to this issue the total distributed reward in a series will be way less than the original `totalRewards` amounts since the number of epochs multiplied by the final reward value will be less than the original documented and set `totalRewards` values.

Recommendation: We recommend reviewing the logic and documenting its rationale accordingly.

QSP-5 Undistributed Epoch Zero Reward

Severity: Medium Risk

Status: Fixed

File(s) affected: `BasePool.sol`, `Pacemaker.sol`

Description: Epoch zero reward will never be distributed since `BasePool.stake` will be allowed to execute only after `starttime` where `starttime` is equal to `HEARTBEATSTARTTIME`.
In the opposite `_currentEpoch` will return zero only if the block timestamp is lower than `HEARTBEATSTARTTIME`.
At any given timestamp the stated condition will not be satisfied simultaneously, meaning that when staking will be allowed the current epoch will be epoch 1.

Recommendation: The logic in `Pacemaker._currentEpoch` should be modified to fix this issue.
Update: The team explained that “epoch zero rewards takes place before Dec 4th, where 2.4 million $HRIMP tokens are distributed to those whose provide liquidity to the LST-ETH Uniswap pool until Dec 4th. Since this distribution does not happen from the contract itself, the Epoch Zero rewards have been removed from the smart contract logic”. From this explanation, we are marking this issue as fixed.

QSP-6 `starttime` Value Is Inconsistent With Specification

Severity: Medium Risk

Status: Fixed

File(s) affected: `Pacemaker.sol`

Description: Following the `$HRIMP` distribution per epoch table as stated in the specification, the start time is set to be December 4th, 2020. However, in `BasePool.sol` (L30), `starttime` is set to `HEARTBEATSTARTTIME`, which in turn, is equal to the `1602288000` timestamp (`2020-10-10 00:00:00 (UTC UTC +00:00)`). Hence, the start time in the code is inconsistent with the spec.

Recommendation: Either change the code s.t. it adheres to the spec, or change the specification s.t. it reflects the code.

QSP-7 Staking Manipulation

Severity: Low Risk

Status: Fixed

File(s) affected: `BasePool.sol`

Description: A user staking right before the end of an epoch will still be able to get the reward for the ongoing reward even if he staked for only one block.

Recommendation: The current staking algorithm does not take into account the staking time for each user, and should be modified to do so if the aforementioned behaviour is not desired.

QSP-8 Input Validation

Severity: Low Risk

Status: Fixed

File(s) affected: `LPTokenWrapper.sol`, `BasePool.sol`

Description: The address parameters received as input parameters do not have any sanity checks. Example: in the constructor of `LPTokenWrapper.sol`, the `lpTokenAddress` could be `0x0`. If so, the underlying `lpToken` will also be `0x0`, which will lead to all operations dependent on it to fail.

Recommendation: Add checks verifying that the given input addresses are not `0x0`. If there is any expectation that any of those addresses should also be a contract, add a corresponding check.

QSP-9 Incorrect Reward Per Stake

Severity: Medium Risk

Status: Acknowledged

File(s) affected: `BasePool.sol`

Description: When updating `cachedRewardPerStake` in `updateRewards` modifier, the calculated value might be wrong. As example, if we assume that `updateRewards` was not called between epoch X and Y (X<Y) and where X belongs to a different rewards series than Y, the distributed reward will vary from the expected total reward distribution depending on how far apart are X and Y epochs.
This issue is due to how `rewardPerStake` is implemented since `lastUpdateTime` was in a previous series with a higher or lower reward, and where the actual block.timestamp belongs to an ongoing series. The `rewardRate(currentEpoch())` will return the value of the new series and apply it to a period of the previous series distributing more or less reward than expected to the users.

Recommendation: The different series intervals must be taken into account when calculating the cached reward per stake.
Update: The Team will ensure that our users are informed about this issue. We will also employ methods to update both the `cachedRewardPerStake` and `lastUpdateTime` values directly from the UI just before each series ends.

Automated Analyses

Slither

Slither analysis was executed successfully. All raised issues were classified as false positives.

Adherence to Best Practices

- **[Fixed]** Remove the comment in `contracts/heartbeat/Pacemaker.sol` (L18), as it could confuse readers.
- **[Fixed]** We recommend renaming the following variables to improve readability: `HEARTBEATSTARTTIME` => `HEART_BEAT_START_TIME`, `EPOCHPERIOD` => `EPOCH_PERIOD`, `WARMUPPERIOD` => `WARMUP_PERIOD`, `HALFLIFE` => `HALF_LIFE`
- **[Fixed]** `WARMUPPERIOD` is not used anywhere. Consider removing it.
- **[Fixed]** In `contracts/farming/BasePool.sol`, `starttime` is not in lower camel case, as other variables. We suggest following conventions already defined in most of the code.
- **[Fixed]** In `BasePool.sol`, improve error message on L51.
- **[Fixed]** `Pacemaker` variables in L19-L21 do not have any visibility set.

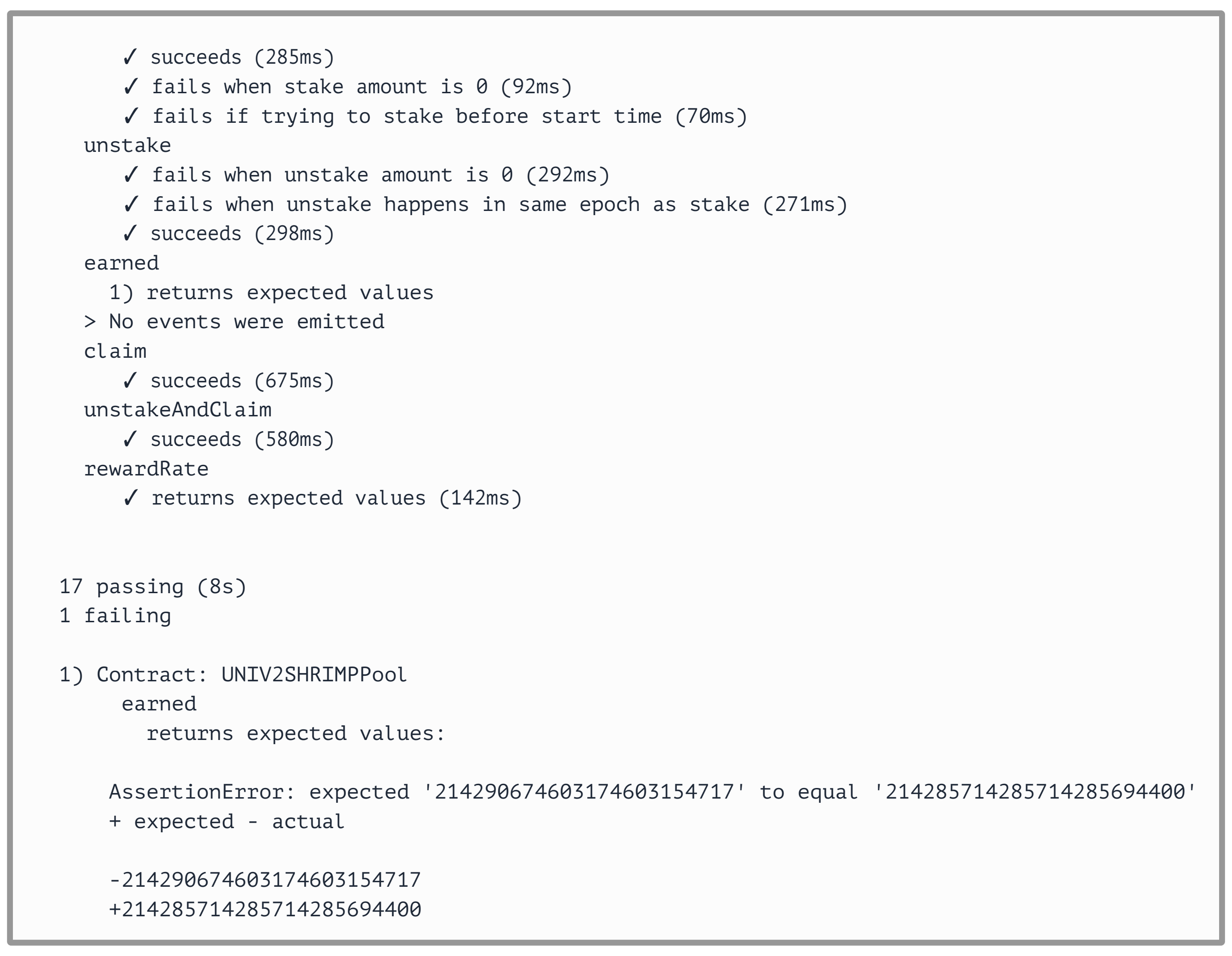
Test Results

Test Suite Results

All tests were successful except one. The user earning output is sensitive to the block timestamp therefore the test output cannot be predicted. The test can be workaround by using an acceptable range for the output.

```
Contract: Pacemaker
  constructor
     ✓ deploys with owner (48ms)
  currentEpoch
     ✓ check currentEpoch after starttime (51ms)
     ✓ check currentEpoch before starttime (58ms)
  epochStartTimeFromTimestamp
     ✓ returns correct value for currentEpoch (97ms)
  epochEndTimeFromTimestamp
     ✓ returns correct value for currentEpoch (162ms)

Contract: UNIV2SHRIMPPool
  constructor
     ✓ fails when deployed with invalid rewardTokenAddress (501ms)
     ✓ fails when deployed with invalid lpTokenAddress (424ms)
     ✓ deploys successfully (145ms)
  stake
```



```
      ✓ succeeds (285ms)
      ✓ fails when stake amount is 0 (92ms)
      ✓ fails if trying to stake before start time (70ms)
    unstake
      ✓ fails when unstake amount is 0 (292ms)
      ✓ fails when unstake happens in same epoch as stake (271ms)
      ✓ succeeds (298ms)
    earned
      1) returns expected values
    > No events were emitted
    claim
      ✓ succeeds (675ms)
    unstakeAndClaim
      ✓ succeeds (580ms)
    rewardRate
      ✓ returns expected values (142ms)


  17 passing (8s)
  1 failing

  1) Contract: UNIV2SHRIMPPool
       earned
         returns expected values:

      AssertionError: expected '2142906746031746031547 17' to equal '2142857142857142856944 00'
      + expected - actual

      -21429067460317460315471 7
      +21428571428571428569440 0
```

Code Coverage

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
farming/	100	100	100	100	
BasePool.sol	100	100	100	100	
LPTokenWrapper.sol	100	100	100	100	
UNIV2SHRIMPPool.sol	100	100	100	100	
heartbeat/	100	100	100	100	
Pacemaker.sol	100	100	100	100	
mocks/	100	100	100	100	
Mock$ HRIMP.sol	100	100	100	100	
MockERC20.sol	100	100	100	100	
MockLSTWETHUNIV2.sol	100	100	100	100	
MockPacemaker.sol	100	100	100	100	
All files	**100**	**100**	**100**	**100**	

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Contracts

```
f40aa41e05fd8c3aeb67aed3ae8c301336f5607fc8cb55421c62e807f3f70287 ./contracts/farming/BasePool.sol
5cc84dafae12cdbd643ce8dec0631b2ba6a693d606ed9f296e37df548b2663d0 ./contracts/farming/LPTokenWrapper.sol
8da6aa4dc86ac0fe293b775dc83c571281190e33f7c708d2697939d25c7d1b62 ./contracts/farming/UNIV2SHRIMPPool.sol
3812aadabb42390c763280c930d301132934b88b2e362197aa723e85f9b61b7a ./contracts/heartbeat/Pacemaker.sol
```

Tests

```
f444f9851b4b4638be774fb1ef241478cc3810bed936c4e8422c60034bf7fa3d ./test/contracts.test.js
255056fef53fa20922d58a6e1e92f52219c05bf712c0e7fd9314a147162cec21 ./test/helpers/currentEpoch.js
a41ae5b2133f64cf5b86abe5ef2d8cb66d77a910161517d49b42df83926b766e ./test/helpers/$HRIMPRewardsCalculator.js
9df013771d7581a713eeabed01731aa0827d5a6403bfa919c81ea533304caacc ./test/farming/test.UNIV2SHRIMPPool.js
8d069fae9c543f2661a16a6ebced8ad42b65b4e2b693f7063287f2533ff6995d ./test/heartbeat/test.Pacemaker.js
```

Changelog

- 2020-11-26 - Initial report
- 2020-11-27 - report review
- 2020-12-04 - reaudit base on commit e32d1306
- 2020-12-07 - Lendroid update on "Incorrect Reward Per Stake"
- 2020-12-08 - Updating tests and coverage result.

About Quantstamp

Quantstamp is a Y Combinator-backed company that helps to secure blockchain platforms at scale using computer-aided reasoning tools, with a mission to help boost the adoption of this exponentially growing technology.

With over 1000 Google scholar citations and numerous published papers, Quantstamp's team has decades of combined experience in formal verification, static analysis, and software verification. Quantstamp has also developed a protocol to help smart contract developers and projects worldwide to perform cost-effective smart contract security scans.

To date, Quantstamp has protected $5B in digital asset risk from hackers and assisted dozens of blockchain projects globally through its white glove security assessment services. As an evangelist of the blockchain ecosystem, Quantstamp assists core infrastructure projects and leading community initiatives such as the Ethereum Community Fund to expedite the adoption of blockchain technology.

Quantstamp's collaborations with leading academic institutions such as the National University of Singapore and MIT (Massachusetts Institute of Technology) reflect our commitment to research, development, and enabling world-class blockchain security.

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp, Inc. (Quantstamp). Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on the website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.

Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.