Quantstamp Security Assessment Certificate

November 21st 2020 — Quantstamp Verified

Linkswap

This security assessment was prepared by Quantstamp, the leader in blockchain security

Executive Summary

Type	Automated Market Maker Contracts
Auditors	Ed Zulkoski, Senior Security Engineer Poming Lee, Research Engineer
Timeline	2020-10-28 through 2020-11-12
EVM	Muir Glacier
Languages	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	None
Documentation Quality	Medium
Test Quality	Undetermined

Source Code

Repository	Commit
None	None

Goals

- Can tokens be locked or stolen?
- Are oracles used properly?
- Do the contracts adhere to best practices and align with inline documentation?

Total Issues	**10** (7 Resolved)
High Risk Issues	0 (0 Resolved)
Medium Risk Issues	**1** (1 Resolved)
Low Risk Issues	**3** (3 Resolved)
Informational Risk Issues	**3** (1 Resolved)
Undetermined Risk Issues	**3** (2 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

During the audit a total of 10 different issues were found of varying severity. Several issues relate to unclear specifications for certain functions, thus requiring further clarification. We suggest that all issues are addressed before using the code in production.
**Update: all issues have been resolved as of recent updates.

ID	Description	Severity	Status
QSP-1	Arbitrary token burning	^ Medium	Fixed
QSP-2	Oracle can never miss an interval, else the `consult` function will experience outages	⌄ Low	Fixed
QSP-3	Missing access control on `purchaseYfl`	⌄ Low	Fixed
QSP-4	`SAME_VOTE` check in `vote` function has false positives	⌄ Low	Fixed
QSP-5	Unlocked Pragma	○ Informational	Fixed
QSP-6	Unchecked function parameters	○ Informational	Acknowledged
QSP-7	Clone-and-Own	○ Informational	Acknowledged
QSP-8	Moving-simple-average over window may lead to arbitrage if `windowSize` is large	? Undetermined	Acknowledged
QSP-9	Unclear functionality in `purchaseYfl`	? Undetermined	Fixed
QSP-10	Unsafe External Call in `function purchaseYfl`	? Undetermined	Fixed

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 v0.6.7
- Mythril v0.2.22

Steps taken to run the tools:

1. Installed the Slither tool: `pip install slither-analyzer`
2. Run Slither from the project directory: `slither .`
3. Installed the Mythril tool from Pypi: `pip3 install mythril`
4. Ran the Mythril tool on each contract: `myth -x path/to/contract`

Findings

QSP-1 Arbitrary token burning

Severity: *Medium Risk*

Status: Fixed

File(s) affected: `LinkswapPair.sol`

Description: The function `burn` has no access restriction. Anyone can use this function to burn all the balance holds by this contract.

Recommendation: Add proper access control to `burn`.
Update: This functionality behaves as in Uniswap V2 here. There will only be tokens to burn if pair tokens are transferred to the contract manually and not via a router.

QSP-2 Oracle can never miss an interval, else the `consult` function will experience outages

Severity: *Low Risk*

Status: Fixed

File(s) affected: `LinkswapPriceOracle.sol`

Description: The function `getFirstObservationInWindow` returns the observation from the oldest epoch, which is set when `update` is invoked earlier in the `window`. However, if for any reason an epoch is missed by the oracle (e.g., via server outages, congested network, etc.), the epoch will continue to have stale data. If `consult` is invoked when this stale data is returned by `getFirstObservationInWindow`, the function will revert for the entire current epoch. Note that this effects upstream functions including `calculateTokenAmountFromUsdAmount`.

Recommendation: Ensure that the oracle can reliably check in every epoch.

QSP-3 Missing access control on `purchaseYfl`

Severity: *Low Risk*

Status: Fixed

File(s) affected: `YFLPurchaser.sol`

Description: The function is `external`, however all tokens transferred to this contract and then used to purchase YFL are sent to the governance contract, not the original `msg.sender`.

Recommendation: Restrict access control of `purchaseYfl` such that it can only be invoked from the governance contract.

QSP-4 `SAME_VOTE` check in `vote` function has false positives

Severity: *Low Risk*

Status: Fixed

File(s) affected: `yYFL.sol`

Description: On L165: `require(support != (proposal.forVotes[msg.sender] > 0), "yYFL: SAME_VOTE");`, the conditional may incorrectly report a "same vote" when the vote has changed.
Consider the following scenario:

1. A user has 10 YFL tokens that they use to vote for Proposal A, locking all 10 tokens;
2. The user gains 10 more tokens, and votes on Proposal B with all 20 tokens, also increasing their lock to 20.
3. The user wishes to increase their vote to 20 tokens for Proposal A. Both checks on L164-165 will succeed, triggering the revert.

Recommendation: Check the `voteAmount` against either `proposal.forVotes` or `proposal.againstVotes` (instead of against `voteLockAmount[msg.sender]`) for the `msg.sender`

QSP-5 Unlocked Pragma

Severity: *Informational*

Status: Fixed

File(s) affected: `All Contracts`

Description: Every Solidity file specifies in the header a version number of the format `pragma solidity (^)0.4.*`. The caret (^) before the version number implies an unlocked pragma, meaning that the compiler will use the specified version *and above*, hence the term "unlocked".

Recommendation: For consistency and to prevent unexpected behavior in the future, it is recommended to remove the caret to lock the file onto a specific Solidity version.

QSP-6 Unchecked function parameters

Severity: *Informational*

Status: Acknowledged

File(s) affected: `LinkswapFactory.sol`, `LinkswapPriceOracle.sol`, `YFLPurchaser.sol`, `yYFL.sol`, `LinkswapLibrary.sol`, `LinkswapPair.sol`, `SafeERC20Namer.sol`, `UQ112x112.sol`

Description:

1. In order to avoid faulty transactions that use default values for arguments, we recommend checking that address arguments are non-zero in the following functions:
    1. `LinkswapFactory.createPair`;
    2. `LinkswapPriceOracle.constructor`;
    3. `YFLPurchaser.constructor`;
    4. `yYFL.constructor`.
2. For the functions `getAmountOut` and `getAmountIn` in `LinkswapLibrary.sol`, it should check if the input variable `tradingFeePercent` is smaller than `1e6`.
3. For the function `getAmountsOut` and `getAmountsIn` in `LinkswapLibrary.sol`, it should check if each of the array element in the input variable `path` is not equal to `0x0`.
4. In function `swap` of `LinkswapPair.sol`, it should check if `to` is a `ILinkswapCallee` contract.
5. Although the function `parseStringData` in `SafeERC20Namer.sol` appears not to be used anywhere in this project, it should check if `b` contains more than `64` bytes, and also should check if `b` contains exactly `charCount + 64` bytes.
6. On `L18` in `UQ112x112.sol`, it should check if `uint112 y` is not equal to `0`.

Recommendation: Add `require` statements for to each of the above functions.
Update: The constructor checks have been added. Other checks have not been added following the precedent of Uniswap contracts.

QSP-7 Clone-and-Own

Severity: *Informational*

Status: Acknowledged

File(s) affected: `SafeMathLinkswap.sol`

Description: The clone-and-own approach involves copying and adjusting open source code at one's own discretion. From the development perspective, it is initially beneficial as it reduces the amount of effort. However, from the security perspective, it involves some risks as the code may not follow the best practices, may contain a security vulnerability, or may include intentionally or unintentionally modified upstream libraries.
In particular, the file `SafeMathLinkswap.sol` appears to be cloned from another project.

Recommendation: Rather than the clone-and-own approach, a good industry practice is to use the Truffle framework for managing library dependencies. This eliminates the clone-and-own risks yet allows for following best practices, such as, using libraries.
Update: This file is cloned from Uniswap here.

QSP-8 Moving-simple-average over window may lead to arbitrage if `windowSize` is large

Severity: *Undetermined*

Status: Acknowledged

File(s) affected: `LinkswapPriceOracle.sol`

Description: If the price has changed significantly over the `windowSize`, older values may not be helpful at determining the current value of a coin. For example, if the average price has significantly increased over 24h, then `consult` will return a higher `out` amount than desired (as the older low values will lower the average price).

Recommendation: Ensure that the window size is not overly large to prevent stale data from being used. If it is desirable to weigh recent data more than older data, an exponential moving average (EMA) could be used instead of a simple average.
Update from the Linkswap team: The price oracle is only used for calculating listing fees. Calculations based on slightly outdated prices is acceptable.

QSP-9 Unclear functionality in `purchaseYfl`

Severity: *Undetermined*

Status: Fixed

File(s) affected: `YFLPurchaser.sol`

Description: The function accepts a list of tokens, however conversions are only made between the pairs `(LINK, WETH)` and `(WETH, YFL)` Any other tokens are simply sent back to the governance address. Since the function is not documented, it is unclear what the intended semantics should be.

Recommendation: Add documentation to this function. Disallow tokens that are not actually used for conversion.

QSP-10 Unsafe External Call in `function purchaseYfl`

Severity: *Undetermined*

Status: Fixed

File(s) affected: `YFLPurchaser.sol`

Description: An attacker can execute arbitrary code by providing a fake token address to the function `purchaseYfl`.

Recommendation: Add `reentrancyLock` to this function.

Automated Analyses

Slither

1. The return value of the following function calls should be checked:
    1. `LinkswapRouter.createPairUsingETH` ignores return value by `IERC20(WETH).approve(factory,msg.value)`
    2. `LinkswapRouter.removeLiquidity` ignores return value by `IERC20(pair).transferFrom(msg.sender,pair,liquidity)`

2. In `yYFL.executeProposal`, `proposal.executed = true` should be set before the e

Mythril

Mythril did not report any issues, however it was limited by using argument `t 1` due to performance issues.

Code Documentation

1. The function `AddressStringUtil.toAsciiString` "converts an address to the uppercase hex string, extracting only len bytes (up to 20, multiple of 2)". It should be mentioned that the `len` parameter is the number of nibbles, not bytes though. It should also mention that if `len < 40`, the *left-most* bytes are the ones extracted. **Update:** this follows precedent of Uniswap here.
2. On L157 of `LinkswapFactory.sol`: `address lockupToken, // LINK or WETH`, the comment is not entirely accurate, since this may be a different token if the pair has been pre-approved by governance. **Update:** fixed.
3. In `LinkswapPriceOracle.sol` the comments on L34: `[windowSize - (windowSize / granularity) * 2, windowSize]` and L37: `[now - [22 hours, 24 hours], now]` are not fully clear. Why is there a `* 2` in the equation? **Update:** this follows precedent of Uniswap [here](https://github.com/Uniswap/uniswap-v2-periphery/blob/master/contracts/examples/ExampleSlidingWindowOracle.sol).
4. In `FixedPoint.sol`, it is unclear why is the `<< 56` is needed in L72. **Update:** this follows precedent of Uniswap here.
5. It is unclear if the formula in L179-L180 in function `_mintFee` in `LinkswapPair.sol` is correctly implemented. There is no explanation for the formula. **Update:** this follows precedent of Uniswap here.

Adherence to Best Practices

1. On L39 of `LinkswapFactory.sol`, the `unlocked` variable could be a boolean, however the standard Reentrancy Guard library could be reused instead. **Update:** fixed.
2. The reentrancy logic mentioned above is cloned in several places such as `yYFL.sol`. **Update:** fixed.
3. In `LinkswapPriceOracle.sol`, the assignment and comparison on L68: `(periodSize = _windowSize / _granularity) * _granularity == _windowSize` should be split across multiple lines. **Update:** this follows the precedent of Uniswap [here](https://github.com/Uniswap/uniswap-v2-periphery/blob/master/contracts/examples/ExampleSlidingWindowOracle.sol).
4. The files `Babylonian.sol` and `Math.sol` are clones of each other. **Update:** fixed.
5. Favor using `uint256` instead of `uint`. **Update:** fixed.
6. In `L12` in the function `sortTokens` in `LinkswapLibrary.sol` it is comparing the magnitude of the addresses; is this intended? **Update:** this follows the precedent of Uniswap [here](https://github.com/Uniswap/uniswap-v2- periphery/blob/master/contracts/libraries/UniswapV2Library.sol).
7. In the function `getReserves` in `LinkswapLibrary.sol`, the sort is done based on the addresses instead of the reserves, is this intended? **Update:** this follows the precedent of Uniswap [here](https://github.com/Uniswap/uniswap-v2- periphery/blob/master/contracts/libraries/UniswapV2Library.sol).
8. In `YFLPurchaser.sol`, when `tokens[i] == link`, it will never transfer the purchased YFL token to the governance contract, is this intended? **Update:** fixed.

Test Results

Test Suite Results

```
AddressStringUtil
  #toAsciiString
     ✓ zero address (83ms)
     ✓ own address (47ms)
     ✓ random address (40ms)
     ✓ reverts if len % 2 != 0
     ✓ reverts if len >= 40
     ✓ reverts if len == 0
     ✓ produces len characters (64ms)

Babylonian
  #sqrt
     ✓ works for 0-99 (1206ms)
     ✓ max uint256

FixedPoint
  #encode
     ✓ shifts left by 112
     ✓ will not take >uint112(-1)
  #encode144
     ✓ shifts left by 112
     ✓ will not take >uint144(-1)
  #decode
     ✓ shifts right by 112
     ✓ will not take >uint224(-1)
  #decode144
     ✓ shifts right by 112
     ✓ will not take >uint256(-1)
  #div
     ✓ correct division
     ✓ throws for div by zero
  #mul
     ✓ correct multiplication
     ✓ overflow
     ✓ max of q112x112
  #fraction
     ✓ correct computation less than 1
     ✓ correct computation greater than 1
     ✓ fails with 0 denominator
  #reciprocal
     ✓ works for 0.25
     ✓ fails for 0
     ✓ works for 5
  #sqrt
     ✓ works for 25
     ✓ works with numbers less than 1
     ✓ works with 0

PairNamer
  #pairName
     ✓ concatenation (100ms)
```

```
  #pairSymbol
    ✓ concatenation (85ms)

SafeERC20Namer
  #tokenName
    ✓ works with compliant (61ms)
    ✓ works with noncompliant (59ms)
    ✓ works with empty bytes32 (65ms)
    ✓ works with noncompliant full bytes32 (69ms)
    ✓ works with optional (56ms)
    ✓ works with non-code address
    ✓ works with really long strings (83ms)
    ✓ falls back to address with empty strings (59ms)
  #tokenSymbol
    ✓ works with compliant (52ms)
    ✓ works with noncompliant (53ms)
    ✓ works with empty bytes32 (52ms)
    ✓ works with noncompliant full bytes32 (71ms)
    ✓ works with optional (43ms)
    ✓ works with non-code address
    ✓ works with really long strings (86ms)
    ✓ falls back to address with empty strings (51ms)

TransferHelper
  #safeApprove
    ✓ succeeds with compliant with no revert and true return (94ms)
    ✓ fails with compliant with no revert and false return (53ms)
    ✓ fails with compliant with revert (60ms)
    ✓ succeeds with noncompliant (no return) with no revert (175ms)
    ✓ fails with noncompliant (no return) with revert (56ms)
  #safeTransfer
    ✓ succeeds with compliant with no revert and true return (101ms)
    ✓ fails with compliant with no revert and false return (52ms)
    ✓ fails with compliant with revert (58ms)
    ✓ succeeds with noncompliant (no return) with no revert (89ms)
    ✓ fails with noncompliant (no return) with revert (127ms)
  #safeTransferFrom
    ✓ succeeds with compliant with no revert and true return (80ms)
    ✓ fails with compliant with no revert and false return (55ms)
    ✓ fails with compliant with revert (60ms)
    ✓ succeeds with noncompliant (no return) with no revert (85ms)
    ✓ fails with noncompliant (no return) with revert (55ms)
  #safeTransferETH
    ✓ succeeds call not reverted (171ms)
    ✓ fails if call reverts (54ms)

LinkswapERC20
  ✓ name, symbol, decimals, totalSupply, balanceOf, DOMAIN_SEPARATOR, PERMIT_TYPEHASH (67ms)
  ✓ approve (66ms)
  ✓ transfer (76ms)
  ✓ transfer:fail
  ✓ transferFrom (131ms)
  ✓ transferFrom:max (115ms)
  ✓ permit (104ms)

LinkswapFactory
  ✓ initial values (193ms)
  ✓ LinkswapPair bytecode hash
  ✓ approvePairViaGovernance:notGovernance
  ✓ approvePairViaGovernance:identicalAddresses
  ✓ approvePairViaGovernance:zeroAddress
  ✓ approvePairViaGovernance:success (207ms)
  ✓ approvePairViaGovernance:reverse (205ms)
  ✓ createPairViaGovernance (970ms)
  ✓ createPair:identicalAddresses
  ✓ createPair:zeroAddress (41ms)
  ✓ createPair:governance
  ✓ createPair:linkPair (1028ms)
  ✓ createPair:linkWethPair (1159ms)
  ✓ createPair:wethPair (995ms)
  ✓ createPair:linkPair:feeChanged (1026ms)
  ✓ createPair:nonLinkPair:feeChanged (1318ms)
  ✓ createPair:linkListingFee (1245ms)
  ✓ createPair:wethListingFee (1142ms)
  ✓ createPair:yflListingFee (1119ms)
  ✓ createPair:listingFee:split (1147ms)
  ✓ createPair:transferFromFailed (1046ms)
  ✓ createPair:invalidPair (44ms)
  ✓ createPair:invalidListingFeeToken
  ✓ createPair:invalidListingLockupAmount:link (104ms)
  ✓ createPair:invalidListingLockupAmount:weth (100ms)
  ✓ createPair:invalidListingLockupPeriod (102ms)
  ✓ createPair:lockup:zeroListingFee (2223ms)
  ✓ createPair:lockup:withDiscountedListingFee (2543ms)
  ✓ setPriceOracle (116ms)
  ✓ setTreasury (113ms)
  ✓ setGovernance (75ms)
  ✓ setTreasuryProtocolFeeShare (249ms)
  ✓ setProtocolFeeFractionInverse (282ms)
  ✓ setLinkListingFeeInUsd (185ms)
  ✓ setWethListingFeeInUsd (188ms)
  ✓ setYflListingFeeInUsd (184ms)
  ✓ setTreasuryListingFeeShare (296ms)
  ✓ setMinListingLockupAmountInUsd (401ms)
  ✓ setTargetListingLockupAmountInUsd (203ms)
  ✓ setTargetListingLockupAmountInUsd:lessThanMin (94ms)
  ✓ setMinListingLockupPeriod (244ms)
  ✓ setTargetListingLockupPeriod (185ms)
  ✓ setTargetListingLockupPeriod:lessThanMin (110ms)
  ✓ setLockupAmountListingFeeDiscountShare (314ms)
  ✓ setDefaultLinkTradingFeePercent (247ms)
  ✓ setDefaultNonLinkTradingFeePercent (182ms)
  ✓ setMaxSlippagePercent (347ms)
  ✓ setMaxSlippageBlocks (227ms)

LinkswapPair
  ✓ lock:insufficientLiquidity (147ms)
  ✓ lock:lockAgain:zeroPeriodZeroAmount (334ms)
  ✓ lock:lockAgain:zeroPeriodPositiveAmount (579ms)
  ✓ lock:lockAgain:positivePeriodZeroAmount (479ms)
  ✓ lock:lockAgain:positivePeriodPositiveAmount (469ms)
  ✓ lock:zeroPeriod
  ✓ lock:zeroAmount
  ✓ lock:zeroPeriodZeroAmount (270ms)
  ✓ unlock:alreadyUnlocked (176ms)
  ✓ unlock:beforeExpiry (247ms)
  ✓ lock/unlock:allLiquidity (364ms)
  ✓ lock/unlock:partialLiquidity (359ms)
  ✓ mint (196ms)
  ✓ getInputPrice:0 (280ms)
  ✓ getInputPrice:1 (266ms)
  ✓ getInputPrice:2 (276ms)
  ✓ getInputPrice:3 (254ms)
  ✓ getInputPrice:4 (265ms)
  ✓ getInputPrice:5 (273ms)
  ✓ getInputPrice:6 (255ms)
  ✓ optimistic:0 (292ms)
  ✓ optimistic:1 (267ms)
  ✓ optimistic:2 (265ms)
  ✓ optimistic:3 (252ms)
  ✓ swap:feeChanged (364ms)
  ✓ swap:sliplock:tooMuchUpwardSlippage (422ms)
  ✓ swap:sliplock:maxUpwardSlippage (1330ms)
  ✓ swap:sliplock:maxUpwardSlippage:afterReset (1337ms)
  ✓ swap:sliplock:tooMuchDownwardSlippage (426ms)
  ✓ swap:sliplock:downwardSlippage (1289ms)
  ✓ swap:sliplock:downwardSlippage:afterReset (1195ms)
  ✓ swap:token0 (284ms)
  ✓ swap:token1 (289ms)
  ✓ burn (302ms)
  ✓ price{0,1}CumulativeLast (449ms)
  ✓ protocolFee:off (442ms)
  ✓ protocolFee:on (490ms)
  ✓ protocolFee:on:changedAndSplit (629ms)
  ✓ setTradingFeePercent (307ms)

LinkswapPriceOracle
  #calculateTokenAmountFromUsdAmount
    ✓ link token
    ✓ weth token
```

```
      ✓ max uint
      ✓ unexpected token (53ms)
    yfl token
       ✓ success (105ms)
       ✓ yfl token: no oracle update
       ✓ yfl token: only one oracle update
  #calculateUsdAmountFromTokenAmount
     ✓ link token
     ✓ weth token
     ✓ max uint
     ✓ unexpected yfl token
     ✓ unexpected token (49ms)

LinkswapRouter
   ✓ createPairUsingETH:wethPair (227ms)
   ✓ createPairUsingETH:ethListingFee (325ms)
   ✓ createPairUsingETH:ethPairAndEthListingFee (252ms)
   ✓ quote (43ms)
   ✓ getAmountOut (41ms)
   ✓ getAmountIn
   ✓ getAmountsOut (169ms)
   ✓ getAmountsIn (174ms)

fee-on-transfer tokens
   ✓ removeLiquidityETHSupportingFeeOnTransferTokens (358ms)
   ✓ removeLiquidityETHWithPermitSupportingFeeOnTransferTokens (283ms)
   ✓ swapExactETHForTokensSupportingFeeOnTransferTokens (165ms)
   ✓ swapExactTokensForETHSupportingFeeOnTransferTokens (206ms)
  swapExactTokensForTokensSupportingFeeOnTransferTokens
     ✓ DTT -> WETH (101ms)
     ✓ WETH -> DTT (160ms)

fee-on-transfer tokens: reloaded
  swapExactTokensForTokensSupportingFeeOnTransferTokens
     ✓ DTT -> DTT2 (102ms)
  swapExactTokensForTokensSupportingFeeOnTransferTokens
     ✓ DTT -> DTT2 (102ms)

yYFL
   ✓ initial values (101ms)
   ✓ stake:zero
   ✓ stake:insufficientYfl
   ✓ stake:success (1055ms)
   ✓ stake:maintainYflValue (2006ms)
   ✓ getPricePerFullShare:zero
   ✓ getStakeYflValue:zero
   ✓ withdraw:zero (418ms)
   ✓ withdraw:insufficientBalance (384ms)
   ✓ withdraw:checkVoteExpiry (1644ms)
   ✓ withdraw:early (841ms)
   ✓ withdraw:notEarly (564ms)
   ✓ withdraw:partial (794ms)
   ✓ withdraw:voteLock (1868ms)
   ✓ propose:arityMismatch (85ms)
   ✓ propose:noActions
   ✓ propose:tooManyActions (39ms)
   ✓ propose:insufficientYflForProposal (405ms)
   ✓ propose:success (1149ms)
   ✓ propose:pricePerFullShareAboveOne (1096ms)
   ✓ propose:hasActiveProposal (1136ms)
   ✓ vote:invalidProposalId
   ✓ vote:votingEnded (1131ms)
   ✓ vote:zero (1038ms)
   ✓ vote:insufficientBalance (1085ms)
   ✓ vote:smallerVote (1415ms)
   ✓ vote:sameVote (1490ms)
   ✓ vote:for:full (1445ms)
   ✓ vote:for:partial (1687ms)
   ✓ vote:for:changeSupport (1741ms)
   ✓ vote:checkVoteExpiry (2898ms)
   ✓ executeProposal:invalidProposalId
   ✓ executeProposal:proposalInVoting (1104ms)
   ✓ executeProposal:proposalDidNotPass (1803ms)
   ✓ executeProposal:expired (1754ms)
   ✓ executeProposal:failedExecution (1643ms)
   ✓ executeProposal:success (1845ms)
   ✓ executeProposal:multiple (1921ms)
   ✓ executeProposal:withValue (1328ms)
   ✓ executeProposal:withValue:insufficientEth (1374ms)
   ✓ convertTokensToYfl:invalidYflPurchaser
   ✓ convertTokensToYfl:arityMismatch (1504ms)
   ✓ convertTokensToYfl:alreadyConverted (1435ms)
   ✓ convertTokensToYfl:noYflPurchased (1465ms)
   ✓ convertTokensToYfl (2074ms)
   ✓ setTreasury:forbidden
   ✓ setTreasury (1408ms)
   ✓ setTreasuryEarlyWithdrawalFeeShare:forbidden
   ✓ setTreasuryEarlyWithdrawalFeeShare:0% (1418ms)
   ✓ setTreasuryEarlyWithdrawalFeeShare:12.3456% (1437ms)
   ✓ setTreasuryEarlyWithdrawalFeeShare:100% (1535ms)
   ✓ setTreasuryEarlyWithdrawalFeeShare:>100% (1681ms)
   ✓ setYflPurchaser:forbidden
   ✓ setYflPurchaser:zeroAddress (1778ms)
   ✓ setYflPurchaser (1543ms)
   ✓ setBlocksForNoWithdrawalFee:forbidden
   ✓ setBlocksForNoWithdrawalFee:0 (1684ms)
   ✓ setBlocksForNoWithdrawalFee:345600 (1441ms)
   ✓ setBlocksForNoWithdrawalFee:>345600 (1736ms)
   ✓ setEarlyWithdrawalFeePercent:forbidden
   ✓ setEarlyWithdrawalFeePercent:0% (1506ms)
   ✓ setEarlyWithdrawalFeePercent:100% (1459ms)
   ✓ setEarlyWithdrawalFeePercent:>100% (1891ms)
   ✓ setVotingPeriodBlocks:forbidden
   ✓ setVotingPeriodBlocks:<1920 (1777ms)
   ✓ setVotingPeriodBlocks:1920 (1406ms)
   ✓ setVotingPeriodBlocks:80640 (1405ms)
   ✓ setVotingPeriodBlocks:>80640 (1648ms)
   ✓ setMinYflForProposal:forbidden
   ✓ setMinYflForProposal:<0.01 (1782ms)
   ✓ setMinYflForProposal:0.01 (1406ms)
   ✓ setMinYflForProposal:520 (1396ms)
   ✓ setMinYflForProposal:>520 (1760ms)
   ✓ setQuorumPercent:forbidden
   ✓ setQuorumPercent:<10% (1660ms)
   ✓ setQuorumPercent:10% (1475ms)
   ✓ setQuorumPercent:33% (1423ms)
   ✓ setQuorumPercent:>33% (1821ms)
   ✓ setVoteThresholdPercent:forbidden
   ✓ setVoteThresholdPercent:<50% (1674ms)
   ✓ setVoteThresholdPercent:50% (1391ms)
   ✓ setVoteThresholdPercent:66% (1416ms)
   ✓ setVoteThresholdPercent:>66% (1668ms)
   ✓ setExecutionPeriodBlocks:forbidden
   ✓ setExecutionPeriodBlocks:<1920 (1669ms)
   ✓ setExecutionPeriodBlocks:1920 (1432ms)
   ✓ setExecutionPeriodBlocks:172800 (1496ms)
   ✓ setExecutionPeriodBlocks:>172800 (1665ms)
   ✓ stake:forbidden (1679ms)
   ✓ convertTokensToYfl:reentrancyLock (3637ms)
   ✓ withdraw:forbidden (1640ms)
   ✓ propose:reentrancyLock (2501ms)
   ✓ vote:reentrancyLock (1746ms)
   ✓ executeProposal:reentrancyLock (1821ms)
   ✓ transfer:reentrancyLock (1730ms)
   ✓ approve:reentrancyLock (1733ms)
   ✓ transferFrom:reentrancyLock (1756ms)
   ✓ increaseAllowance:reentrancyLock (1807ms)
   ✓ decreaseAllowance:reentrancyLock (1790ms)
   ✓ factory:approvePairViaGovernance (2467ms)
   ✓ factory:setTreasury (1509ms)


288 passing (3m)
```

Code Coverage

Since the project uses `waffle` and `mocha` instead of `truffle`, we could not evaluate test coverage at this time. We recommend adding instructions to the README regarding how to run coverage.

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

```
d264ef4e0337bfff5cd139bdfaf2582c72c989a98776e21e4c73808cfe3fc378 ./contracts/LinkswapFactory.sol
0c63eff95a1336954b0a09c8eb5ee115e6f0cce4751dbf4c1becb14ab56835ef ./contracts/LinkswapPriceOracle.sol
e357c7b57163e29c7cbb27e1b2d88e575387e61a1889df9fc441c8c62fbfa230 ./contracts/LinkswapRouter.sol
ff7eda569cb053c0f54151acb15e3a68103895e5d064205c67db2224d731a791 ./contracts/yYFL.sol
b627038303962e98ca042f74482b90371e57e45cc0741a43a190bffaa09fa31e ./contracts/YFLink.sol
7437dcab67c8139abf22f03de97e083cff5684988b98a0197de20ce0fe9419af ./contracts/YFLPurchaser.sol
cbcec5878f912f32e457b959acd41cfc745d4bcd8f6a14c0d3567e8e935cb731 ./contracts/LinkswapPair.sol
0af9a0e092523c0965ebe1b1087b9ddd2d221adad678c501491174d1c9c4b2ef ./contracts/interfaces/ILinkswapRouter.sol
874d5c781150883fd834cc5b8c11852223f963a738a33ed4c287537cca4eb23c ./contracts/interfaces/ILinkswapFactory.sol
e05ed908f1c34fba0df176bf8be2d07e6d322cfdb271a6a6b230e779a70692a0 ./contracts/interfaces/IyYFL.sol
181b7ea1a410a2b93b351ec0ab0f2819598aaceee68ee280948107a639d07e5b ./contracts/interfaces/IWETH.sol
9578f3cae9a5dd2a8b0153aa31bf182f2b1114adb3607b61d8e76b09785578c9 ./contracts/interfaces/IChainlinkOracle.sol
35d41e8211b18e7c01519b2a4dad6745a46258359da9161eb3ad740e771f977e ./contracts/interfaces/ILinkswapERC20.sol
8311d94bc5f9d34f1a2f54d84ccd3b34d35737d3cb3448149dd5cdf5c82ee736 ./contracts/interfaces/ILinkswapCallee.sol
baf727e597de6520d7c22fd1855e1013240bc2bbabb8f30326992883856e6150 ./contracts/interfaces/ILinkswapPriceOracle.sol
7ed12f1d996f9c1b7fa46bbd7a66fd029ed5405e90027b396df703c2684c9dfe ./contracts/interfaces/IYFLPurchaser.sol
bfdac8fc0490e8675f534f49917ea70abe46e5c3383fba01e0a16c963888b751 ./contracts/interfaces/IUniswapV2Pair.sol
c450f3e6e4140b307e8d8da7e79730263d398be31283354a7304e6aff1f98ef1 ./contracts/interfaces/ILinkswapPair.sol
b7efa1b37618ae156304a04ad454078349c7bc4e174fec2d8c3a92fbbbec1e2e ./contracts/interfaces/IController.sol
4023f62d3dad61bb6d237b7555415f2c40464fdf21006a8680c4b461bbcb5ba7 ./contracts/libraries/SafeERC20Namer.sol
7d06b7944a7428e8532dfdcc2bcbcd7b7b0c131be30e3fccf5ecd815ebdf450b ./contracts/libraries/AddressStringUtil.sol
9bcf78dfd5a75c8bf87c683950bd465718db4588bba4b790698e91d5c0b90242 ./contracts/libraries/LinkswapLibrary.sol
4fca685d502662d7c37bed984509133af4889b42ac3213f532ef8f611b575394 ./contracts/libraries/FixedPoint.sol
5b5c4c9f8893d3b177ae8a4f2f649834d0f0d8fe98a6d19a2ae719a395f06883 ./contracts/libraries/UQ112x112.sol
b651835bccfe3b4a3729eb67170fcddb6c39514e694f08ed69c36b21995a41b9 ./contracts/libraries/PairNamer.sol
49b08137a697651e153f4542a5e35672818e999eee4316812f9613e668644d0f ./contracts/libraries/Math.sol
b56f69629588a073f3e4645895ae41555928cf62f638a45e4a0ae92104b0d2a8 ./contracts/libraries/TransferHelper.sol
c3a1138c0d3fa415e7ec1c2146180240f57523e50f35665e670a127416795d02 ./contracts/libraries/SafeMathLinkswap.sol
e1c65c033e11f3b2aa4a739763b2e6197e6e8af66dbc90659e26721dad7e1a02 ./contracts/test/BabylonianTest.sol
56364a07cd3b5564b2503783cead9255e6c14bf3ee0618883e67243d9cc31592 ./contracts/test/TransferHelperTest.sol
5658c25913d60dfcc83e0fad95bc93d102105056737b8c80c373c78a5d9246a3 ./contracts/test/DeflatingLinkswapERC20Test.sol
96832ae949ab55c05763c41f77e179fe1a085e9b3dae8cf58733ee2b75d69afe ./contracts/test/LinkswapPriceOracleTest.sol
b5ccaf3e808a00e134756763846f3be898c96761401b787406c2585023cfd790 ./contracts/test/ERC20Test.sol
511048aacd29396e713d810be7d58f57bb69d80b19330c9bbb91f49262446de4 ./contracts/test/LinkswapERC20Test.sol
4e500994a8c97582d455aa20d2f7a7606333c76b420aab43116756764670a2ad ./contracts/test/ChainlinkOracleTest.sol
4895433c0d680523c321bef42b849eae7700e1761b30fdc15a2ad89049b70d77 ./contracts/test/RouterEventEmitterTest.sol
cee7a6b16d7b3edac3201ea9533c7035e7f3951e4142e5de5faa97660772365f ./contracts/test/YFLPurchaserTest.sol
c3387f85ae5ab51e2a671e0b56a6722bf7081055e90353275a9b0952c240eb60 ./contracts/test/SafeERC20NamerTest.sol
95b4e2fb31542077f5454ee5de7619c3e27636147c4d909467fb0b8319c31c2c ./contracts/test/FixedPointTest.sol
41b5228d2f0727fe7fc3ee361dc1a30b389bfe1d8d273af2a3494569d83ca4bd ./contracts/test/PairNamerTest.sol
0af46ebc7db393ae41f2cd9702a2bbcb4176ad753c2aa57652419e015fd495a9 ./contracts/test/WETHTest.sol
864a2b98c33fa00f553cd85f261217a4afdb1b1b70a1e7845e8a7077da18f6d0 ./contracts/test/AddressStringUtilTest.sol
```

Tests

```
33cd1665755d1fc8a1b8e9c4f21547a538bb5ad4895bac86db33fc4dd64197c3 ./test/LinkswapRouter.test.ts
```

```
e734d60a00d5785326a38c141c2d759d937f1dd19c79c3511617859caf3e7751 ./test/yYFL.test.ts
f7b388d9ea07305a2f1a2cf6a039aa7144494dfd86ec175612f2dc17f2742fe8 ./test/LinkswapPriceOracle.test.ts
ce996b1bb9296c8572e184f413aeff29a1c57a9df0f0ebc5950683db91e811af ./test/LinkswapERC20.test.ts
931e4aa0398dc5abff904ba3f4dde7b52a1dc3f25d633b876b290eca440b1bb5 ./test/LinkswapPair.test.ts
8ac407904ef1a25a176c475e2aeabd3acb1660f832a5f91220aa8630bbb34e3f ./test/LinkswapFactory.test.ts
62f58eac0881bcfb5d45c531a651fbfeb1e49c8c9da61ff37638c4cd2c4c0191 ./test/shared/fixtures.ts
7ff82f2bec58baecee0a9693efbfa992e20c5a1efce2e876de33516db0d547d1 ./test/shared/utilities.ts
ba708e8905f9e8af956f62af3bee5754548405a5eb37734c654635904c70d891 ./test/libraries/Babylonian.test.ts
e092954a222096b14fc1fb870b1f4a8472674072753a3d944c35d92b2d97a1d6 ./test/libraries/TransferHelper.test.ts
c6188c4c5aa40c30f25a54d0433faa1f71c90312db13b6f01812656f159aa612 ./test/libraries/SafeERC20Namer.test.ts
364c10dd21d8284c153eedd898f496ea1ae4756f740f52b7ec0f65bfd5b2780e ./test/libraries/FixedPoint.test.ts
2f4d4ff9de12253bb30168941b8b5e9379667793c5b04f51c0bf4d3cfabb0477 ./test/libraries/PairNamer.test.ts
30c1b44283bd97aa35a1b28949bbe9a0254702a9423a52e14303941382690c29 ./test/libraries/AddressStringUtil.test.ts
```

Changelog

- 2020-11-03 - Initial report
- 2020-11-12 - Updated report

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.