Quantstamp Security Assessment Certificate

July 6th 2021 — Quantstamp Verified

MCDEX Arbitrum Integration

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

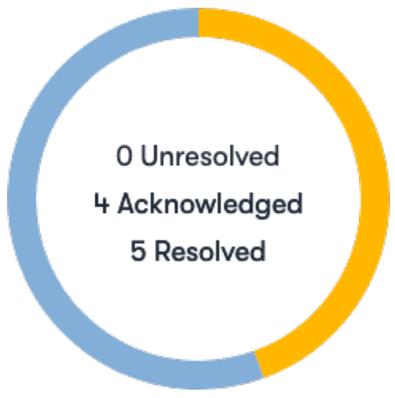
Executive Summary

Type	Decentralized Exchange
Auditors	Jose Ignacio Orlicki, Senior Engineer Jake Goh Si Yuan, Senior Security Researcher Fayçal Lalidji, Security Auditor
Timeline	2021-06-07 through 2021-06-21
EVM	Muir Glacier
Languages	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Reference Document Specification PDF
Documentation Quality	High
Test Quality	Medium

Source Code

Repository	Commit
mai-protocol-v3	50fb550 (initial report)
mai-protocol-v3	edf2053 (reaudit)

Total Issues	9 (5 Resolved)
High Risk Issues	0 (0 Resolved)
Medium Risk Issues	2 (0 Resolved)
Low Risk Issues	4 (2 Resolved)
Informational Risk Issues	2 (2 Resolved)
Undetermined Risk Issues	1 (1 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

We have reviewed the code, documentation, and test suite and found several issues of various severities. Overall, we consider the code to be well-written and with sufficient documentation and an extensive testing suite. We have outlined suggestions to better follow best practices, and recommend addressing all the findings to tighten the contracts for future deployments or contract updates. We also provide suggestions for improvements to follow the best practices. We recommend addressing all the 9 findings and the rest of the suggestions to harden the contracts for future deployments or contract updates. We recommend against deploying the code as-is.

After reaudit: Quantstamp has performed a reaudit to check the proposed fixes. All of the previously identified issues were either resolved (5 issues) or acknowledged (4 issues). Tests are very exhaustive but the output is still very verbose and can be confusing to read. There are 3 tests of 345 tests that are still failing.

ID	Description	Severity	Status
QSP-1	Insurance Fund Not Replenished by Liquidation Penalty Ratio	^ Medium	Acknowledged
QSP-2	No Settlement for Stale Oracle	^ Medium	Acknowledged
QSP-3	Gas Usage / `for` Loop Concerns	⌄ Low	Mitigated
QSP-4	Missing Parameter Validation	⌄ Low	Fixed
QSP-5	Nonce Not Autoincremented	⌄ Low	Acknowledged
QSP-6	Contract Limit Size Exceeded	⌄ Low	Acknowledged
QSP-7	TODO in `Variable.getMCBToken`	○ Informational	Fixed
QSP-8	Implicity Use of Interfaces	○ Informational	Fixed
QSP-9	Rebalance Before Checking Not Fulfilled	? 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.

Findings

QSP-1 Insurance Fund Not Replenished by Liquidation Penalty Ratio

Severity: *Medium Risk*

Status: Acknowledged

File(s) affected: `contracts/module/TradeModule.sol`

Description: There is a recommendation to fill the initial and supplementary Insurance Funds with capital `liquidityPool.donatedInsuranceFund` from the operator (`We encourage operators to donate to the initial capital and supplement the insurance fund as the contract runs.`). There is also a mechanism to refill the Insurance Fund of the Perpetual with a certain ratio (`perpetual.insuranceFundRate` L457 of `TradeModule`) from the liquidation penalty (`When trader's position gets liquidated due to insufficient margin, a certain ratio (based on the AMM parameters) of the charged liquidation penalty goes to the insurance fund`) that goes into `liquidityPool.insuranceFund`. But if the donated operator funds are insufficient then the liquidation penalty fraction assigned to insurance will always be insufficient to bail out the liquidated users because the liquidation penalty is always bigger than the ratio assigned for the replenishment (see L470-473 and check that `penaltyToFund < penalty`). > int256 penaltyToLP = liquidityPool.updateInsuranceFund(penaltyToFund); > perpetual.updateCash(address(this), penaltyToLP); > perpetual.updateCash(liquidator, penaltyToLiquidator); > perpetual.updateCash(trader, penalty.add(vaultFee).neg());

Exploit Scenario: 1. Alice operates a perpetual *P* with `liquidationPenaltyRate = 5%` insuranceFundRate = 20%` . 2. Alice operates perpetual *P* and donated initial insurance capital of 5 DAI. 3. Bob is a user of *P* and creates a position of 100 DAI. 4. Bob is unfortunately liquidated by liquidator Mallory. 5. From the 100 DAI liquidated Bob losses 5% that is 5 DAI. 6. From the 5 DAI in a penalty, Mallory gets 80% that is 4 DAI and the rest 1 DAI goes into the insurance fund. Now the insurance fund has 6 DAI. 7. As the insurance fund can currently cover 5 DAI of Bob loss when Bob removes liquidity of 5 DAI the insurance funds end with 1 DAI. 8. If we go to Step 3 and repeat the process we see that in the second liquidation Bob can only get 1 remaining DAI from the insurance fund and not the 5 DAI to be fully covered.

Recommendation: The insurance fund must be replenished using certain fee ratios from the regular operation of the Perpetual, outside of liquidation scenarios, this way, if actuarial probabilities and liquidation size estimation used are good estimations the users can be assured that they can access an Insured Fund that is not empty. The liquidity of the insurance fund must not come from liquidated amounts but from the regular operation.

Update: According to Project Notes *"Considering the clearing capacity of the blockchain, we have reserved a large margin between "begin to liquidate" and "bankrupt". If the extreme condition happens and the insurance fund decreases to zero, the global settlement begins"*.

QSP-2 No Settlement for Stale Oracle

Severity: *Medium Risk*

Status: Acknowledged

File(s) affected: `contracts/oracle/router/OracleRouter.sol`

Description: According to documentation `when Oracle does not provide updates for over 24 hours, the contract will also enter settlement`, but this functionality is not implemented.

Recommendation: Follow the documentation fallback for Oracle critical problems or document this anomaly.

Update: According to Project Notes *"Already has a check-in OracleWrapper.sol. Rename the folder name to avoid misunderstanding."*.

QSP-3 Gas Usage / `for` Loop Concerns

Severity: *Low Risk*

Status: Mitigated

File(s) affected: `LiquidityPoolModule.sol`

Description: In `LiquidityPoolModule` depending on the number of `perpetuals`, `setAllPerpetualsToEmergencyState` and `addLiquidity` can consume excessive gas.

Recommendation: We recommend running a gas analysis with the most extreme edge case to get an idea about the max gas consumption of the implemented functions.

Update: According to Project Notes *"There is a limit of max 48 perpetuals in one LiquidityPool. "*.

QSP-4 Missing Parameter Validation

Severity: *Low Risk*

Status: Fixed

File(s) affected: `Variables.sol`, `LiquidityPoolModule.sol`, `LiquidityPoolModule.sol`, `LiquidityPoolModule.sol`

Description: 1. Variables.sol::L52 `setVault` can be set to address(0) despite L23 rejecting that.

1. LiquidityPoolModule.sol In `initialize`, should validate `insuranceFundCap >= 0` using at the minimum as a negative cap does not seem to have meaning. Has precedence in `validateLiquidityPoolParameter` and `setLiquidityPoolParameter`.
2. LiquidityPoolModule.sol `setPerpetualOracle` should validate for whether `newOracle != address(0)`. If `address(0)` has a meaning here (ie. no oracle set or not to use oracle), then it should be well documented.
3. LiquidityPoolModule.sol L436 does not check for whether `length > 0`, does this function `setAllPerpetualsToEmergencyState` have meaning when there is no perpetuals?
4. Reader.sol `constructor _poolCreator` against `address(0)` given that it is immutable.

Update: According to Project Notes *"All fixed except 'setPerpetualOracle'. The new oracle address is validated in 'PerpetualModule.sol'."*.

QSP-5 Nonce Not Autoincremented

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `PoolCreator.sol`

Description: `PoolCreator.createLiquidityPool` allows the caller to introduce the same nonce multiple times which will result in a contract deployment failure.

Recommendation: We recommend auto increment the nonce for each caller and remove it from the arguments of the function to avoid the transaction reverting in case of similar nonces for a specific caller.

QSP-6 Contract Limit Size Exceeded

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `LiquidityPoolModule.sol`

Description: Without optimizations, the contract size for `LiquidityPoolModule` several contracts`` exceeds 24576 bytes (a limit introduced in Spurious Dragon hard fork).

Recommendation: Check that with optimizations the final binary contract size is deployable.

Update: According to Project Nots *"Since the contract currently is only deployed on Arbitrum who has a much larger contract size limit, we'll leave it as is"*.

QSP-7 TODO in `Variable.getMCBToken`

Severity: *Informational*

Status: Fixed

Description: We recommend using a mock contract to simulate addresses in a test environment. Never leave TODOs in the contract during the audit phase or deployment phase.

QSP-8 Implicity Use of Interfaces

Severity: *Informational*

Status: Fixed

File(s) affected: `contracts/Perpetual.sol`, `contracts/interface/ILiquidityPool.sol`, `contracts/broker/Broker.sol`

Description: Some interfaces, such as `ILiquidityPool` are used a lot to cast contracts, but the implementation `Perpetual` does not implement explicitly this interface. The implementation assumes implicitly the interface. This can lead to difficulty maintaining the code and error-prone development.

Recommendation: Declare explicitly the inheritance of all interfaces that are used.

QSP-9 Rebalance Before Checking Not Fulfilled

Severity: *Undetermined*

Status: Fixed

File(s) affected: `LiquidityPoolModule.sol`

Description: L178 claims that `isTraderMarginSafe` has to fulfill `the need to rebalance before checking`, but there is no `rebalance` happening here unlike in functions below with the same requirement. A `rebalance()` function is possible missing.

Recommendation: Clarify if documentation or implementation is incorrect.

Code Documentation

1. LiquidityPoolModule.sol::L334 `liquidit` -> `liquidity`
2. LiquidityPoolModule.sol::L209 `initialze` -> `initialize`
3. PoolCreator.sol::L35 `vaersionKey` -> `versionKey`
4. PoolCreator.sol::L52 `vesion` -> `version`
5. Governance.sol::L167 `the oralce contract declares itself as "termainated";` -> the `oracle contract declares itself as "terminated";`
6. Reader.sol::L104 `malform input data` -> `malformed input data`
7. Reader.sol::L150 `amoun` -> `amount`

Adherence to Best Practices

1. VersionControl.sol::L57 `require(!isVersionKeyValid(versionKey), "implementation is already existed");` message is grammatically wrong. Should be "implementation already exists"
2. LiquidityPoolModule::L517 `checkIn` is not just a `check if the operator is alive` as the description writes in L511, but it is a state modifying operation that does not return a boolean. It extends the `operatorExpiration` by `OPERATOR_CHECK_IN_TIMEOUT` which is `10 days` as of time of audit.
3. LiquidityPoolModule.sol::L804 error message is `not all perpetuals are in NORMAL state` which is misleading when it should be `all perpetuals are NOT in NORMAL state`
4. In `CollateralModule` using try-catch with empty `catch {}` is not a common practice because is redundant.

Test Results

Test Suite Results

Two specific tests failed only due to minor `invalid oracle address` and `oracle must be a contract` checks. The remaining 344 tests passed successfully. The test out is too verbose, this cannot be considered a good practice.

```
$ npx hardhat test
Creating simplified Typechain
Successfully generated simplified Typechain artifacts!


  AccessControl
    ✓ privileges (392ms)
```

```
AMM
  isAMMSafe
    ✓ init - ok (51ms)
    ✓ flat - ok
    ✓ short - ok
    ✓ short - fail
    ✓ long - ok
    ✓ long - fail
  getPoolMargin
    ✓ success-0
    ✓ success-1
    ✓ success-2
    ✓ success-3
    ✓ success-4
    ✓ success-5
    ✓ short unsafe (42ms)
    ✓ long unsafe
  getDeltaCash
    ✓ 0 -> +5
    ✓ 0 -> -5
  safePosition
    ✓ init
    ✓ short, infinite max position2, choose max position1 (39ms)
    ✓ short, choose max position1
    ✓ short, choose max position2 (38ms)
    ✓ long, choose max position3
    ✓ zero index price
  trade - success
    ✓ open 0 -> -141.421, near pos2 limit
    ✓ open 0 -> -0.1, effected by spread
    ✓ open -10 -> -141.067, near pos2 limit
    ✓ open -10 -> -10.1, effected by spread
    ✓ open 0 -> 100, near pos2 limit
    ✓ open 0 -> 0.1, effected by spread
    ✓ open 10 -> 100, near pos2 limit
    ✓ open 10 -> 10.1, effected by spread
    ✓ close -10 -> -9, normal
    ✓ open -10 -> -9.9, effected by spread
    ✓ close -10 -> 0, to zero
    ✓ close 10 -> 9, normal
    ✓ close 10 -> 9.9, effected by spread
    ✓ close 10 -> 0
    ✓ close unsafe -10 -> -9, normal
    ✓ close unsafe -10 -> -9.9, small
    ✓ close unsafe 10 -> 9, normal
    ✓ close unsafe 10 -> 9, small
    ✓ close negative price, clip to index*(1-discount)
    ✓ open 0 -> -141.422, partialFill
    ✓ open -10 -> -141.068, pos2 too large, partialFill
    ✓ open -10 already unsafe, partialFill
    ✓ open 0 -> 100.001, partialFill
    ✓ open 10 -> 100.001, partialFill
    ✓ open 10 already unsafe, partialFill
  trade - fail
    ✓ emergency
    ✓ zero trade amount
    ✓ poolMargin = 0
    ✓ open 0 -> -141.422, pos2 too large (41ms)
    ✓ open -10 -> -141.068, pos2 too large
    ✓ open -10 already unsafe
    ✓ open 0 -> 100.001
    ✓ open 10 -> 100.001
    ✓ open 10 already unsafe
  get share to mint
    ✓ init
    ✓ before safe, after safe (44ms)
    ✓ short, before unsafe, after unsafe
    ✓ short, before unsafe, after safe
    ✓ long, before unsafe, after unsafe
    ✓ long, before unsafe, after safe
    ✓ poolMargin = 0 && totalShare != 0
  get cash to add
    ✓ init
    ✓ before safe, after safe
    ✓ short, before unsafe, after unsafe
    ✓ short, before unsafe, after safe
    ✓ long, before unsafe, after unsafe
    ✓ long, before unsafe, after safe
    ✓ poolMargin = 0 && totalShare != 0
  get cash to return
    ✓ no position (39ms)
    ✓ no position, remove all
    ✓ short (38ms)
    ✓ long (41ms)
    ✓ state != NORMAL
    ✓ all states CLEARED (44ms)
    ✓ poolMargin = 0
    ✓ short, before unsafe
    ✓ long, before unsafe
    ✓ short, after unsafe
    ✓ long, after unsafe
    ✓ long, after negative price
    ✓ long, after exceed leverage
    ✓ zero index
    ✓ zero supply of share token
  get share to remove
    ✓ no position (42ms)
    ✓ no position, remove all
    ✓ short (41ms)
    ✓ long (40ms)
    ✓ state != NORMAL
    ✓ all cleared (46ms)
    ✓ poolMargin = 0
    ✓ short, before unsafe
    ✓ long, before unsafe
    ✓ short, after unsafe
    ✓ long, after unsafe
    ✓ long, after negative price
    ✓ long, after exceed leverage
    ✓ zero index
    ✓ zero supply of share token

Order
  ✓ normal

Broker
  ✓ broker (245ms)
  ✓ broker - cancel (174ms)
  ✓ broker - cancel by another signer (210ms)
  ✓ broker - fee (320ms)

Creator
  1) versionControl
  ✓ createLiquidityPoolWith (358ms)
  ✓ tracer (370ms)
  ✓ tracer - 2 (647ms)
  ✓ owner (157ms)

Funding
  updateFundingState
    ✓ state != NORMAL (193ms)
    ✓ init (189ms)
    ✓ current time = liquidityPool time (189ms)
    ✓ normal (199ms)
    ✓ normal (191ms)
  updateFundingRate
    ✓ state != NORMAL (210ms)
    ✓ init (197ms)
    ✓ unsafe (218ms)
    ✓ normal (214ms)
    ✓ exceed limit (224ms)
    ✓ margin < 0 (210ms)

Getter
[ '0x70997970C51812dc3A010C7d01b50e0d17dc79C8' ]
[
  '0x70997970C51812dc3A010C7d01b50e0d17dc79C8',
  '0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC'
]
[
  '0x70997970C51812dc3A010C7d01b50e0d17dc79C8',
  '0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC'
]
[
  '0x70997970C51812dc3A010C7d01b50e0d17dc79C8',
  '0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC'
]
    ✓ main (1046ms)

Governance
  2) "before each" hook for "checkIn"

GovernorAlpha
  ✓ exceptions (42ms)
  ✓ validateProposer (156ms)
  ✓ params (42ms)
```

```
     ✓ quorum
     ✓ quorum critical (114ms)
     ✓ quorum critical - upgrade (855ms)
     ✓ quorum critical - mixed (38ms)
     ✓ quorum critical - mixed (42ms)
Proposal 1
Pending
Active
Succeeded
Queued
MockLiquidityPool :: setFastCreationEnabled(1) called
Executed
     ✓ pass (400ms)
Proposal 1
124555
Active
Active
Defeated
Defeated
Defeated
     ✓ rejected (437ms)
start @ 1127
     ✓ lock (651ms)
start @ 1217
MockLiquidityPool :: setFastCreationEnabled(1) called
     ✓ lock - 2 (446ms)
     ✓ lock - 3 (306ms)
transfer
     ✓ lock - 3 (736ms)
transfer
     ✓ lock - 4 (654ms)
{
  hash: '0xe2d860d0c16740bf1e9550ba051e47c9305d7b5bda5a488d685c74b610b20774',
  type: 0,
  accessList: null,
  blockHash: '0x48627574b4444cd60f691e7c34dbbc8780c07524f76ae51d7bb96eb8a182f134',
  blockNumber: 1498,
  transactionIndex: 0,
  confirmations: 1,
  from: '0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266',
  gasPrice: BigNumber { _hex: '0x01dcd65000', _isBigNumber: true },
  gasLimit: BigNumber { _hex: '0xaf41a8', _isBigNumber: true },
  to: '0xf144d369814090A5c0bFdC7Ab928B20A35D53a10',
  value: BigNumber { _hex: '0x00', _isBigNumber: true },
  nonce: 1421,
  data:
'0x5b195a8200000000000000000000000076e0923b735cbf86b69e7b8fe558886f3840ff1d0000000000000000000000000000000000000000000000000000000016345785d8a0000000000000000000000000000000000000000000000000000000000000b1a2bc2ec500000000000000000000000000000000000000000000000000000000000000
  r: '0x7851bda122b68e21a1cc2588a0bbf3074f11fc41582b2739ec0998e07ba3cb2a',
  s: '0x0cffdf18348538096875a8a37f8e5002797677ca0405e97dd6ccccbdc9a73e28',
  v: 62709,
  creates: null,
  chainId: 31337,
  wait: [Function (anonymous)]
}
0x000000000000000000000000076e0923b735cbf86b69e7b8fe558886f3840ff1d0000000000000000000000000000000000000000000000000000000016345785d8a0000000000000000000000000000000000000000000000000000000000000b1a2bc2ec50000000000000000000000000000000000000000000000000000000000000038d
Active
Succeeded
Queued
Executed
     ✓ create perpetual (1198ms)

  integration - 4 perps, 1 trader. open + close
200000000000000000
share: 0.0
ctk  : 10001.652698605846251062
┌─────────┬──────────────────────────┬──────────────┐
│ (index) │            0             │      1       │
├─────────┼──────────────────────────┼──────────────┤
│    0    │ 'deposit                 ' │ '   336,006' │
│    1    │ 'addLiquidity            ' │ '   535,315' │
│    2    │ 'trade 1 - open          ' │ '   691,562' │
│    3    │ 'trade 2 - open          ' │ '   594,283' │
│    4    │ 'trade 3 - revert        ' │ '   577,057' │
│    5    │ 'trade 4 - close all     ' │ '   459,658' │
│    6    │ 'withdraw                ' │ '   385,316' │
│    7    │ 'removeLiquidity         ' │ '   455,825' │
└─────────┴──────────────────────────┴──────────────┘
     ✓ main (2662ms)
share: 1000.0
ctk  : 9999.999999999
     ✓ main - 6 decimals (1947ms)
0.0
     ✓ main (1580ms)
cash: 0.0 position: 0.0 margin: 0.0 isSafe: true
     ✓ liquidate (904ms)
     ✓ access control (888ms)

  integration2 - 2 perps. special pool states
     ✓ addLiq + trade + removeLiq + liquidate (1140ms)
     ✓ deposit more than balance
     ✓ deposit when not NORMAL
     ✓ add liquidity more than balance
     ✓ add liquidity when not running
     ✓ trade when not authorized (109ms)
     ✓ trade when market closed (136ms)
     ✓ trade when market terminated (118ms)
     ✓ trade when invalid close-only amount (236ms)
     ✓ trade when invalid limit price (168ms)
     ✓ trade when trader unsafe (371ms)
     ✓ trade when exceed open interest limit (142ms)

  integration2 - 2 perps. trade with targetLeverage
[
  BigNumber { _hex: '0x00', _isBigNumber: true },
  BigNumber { _hex: '0x00', _isBigNumber: true },
  BigNumber { _hex: '0x00', _isBigNumber: true },
  BigNumber { _hex: '0x00', _isBigNumber: true },
  BigNumber { _hex: '0x00', _isBigNumber: true },
  true,
  true,
  true,
  BigNumber { _hex: '0x0de0b6b3a7640000', _isBigNumber: true },
  cash: BigNumber { _hex: '0x00', _isBigNumber: true },
  position: BigNumber { _hex: '0x00', _isBigNumber: true },
  availableMargin: BigNumber { _hex: '0x00', _isBigNumber: true },
  margin: BigNumber { _hex: '0x00', _isBigNumber: true },
  settleableMargin: BigNumber { _hex: '0x00', _isBigNumber: true },
  isInitialMarginSafe: true,
  isMaintenanceMarginSafe: true,
  isMarginSafe: true,
  targetLeverage: BigNumber { _hex: '0x0de0b6b3a7640000', _isBigNumber: true }
]
     ✓ get default target leverage
     ✓ addLiq + tradeWithLev long 3, short 2, short 2, long 1 (825ms)
     ✓ deposit + long 3(auto deposit on demand) (283ms)
     ✓ short 1 when MM < margin < IM, normal fees (504ms)
     ✓ short 1 when margin < mm, the profit is large enough, normal fees (578ms)
     ✓ short 1 when mmUnsafe (close positions will cause profit), reduced fees (438ms)
     ✓ short 1 when safe (close positions will cause loss), fees = 0 (574ms)
     ✓ long small amount from 0 position, margin = keeperGasReward (254ms)
     ✓ long 1 amount from 3 position (419ms)
     ✓ long 1 amount from 3 position, margin = initialMargin (501ms)
     ✓ trade on inverse perp (204ms)
     ✓ heart beat stop (384ms)

  integration3 - 2 perps. add/remove liquidity
     ✓ normal case (223ms)

  LibMath
    mostSignificantBit
       ✓ normal
    sqrt
       ✓ small
       ✓ normal

  LibSafeMathExt
    mul
       ✓ uint256 half up
       ✓ int256 half up
       ✓ int256 ceil
       ✓ int256 floor
    div
       ✓ uint256 half up
       ✓ int256 half up
       ✓ int256 ceil
       ✓ int256 floor
    frac
       ✓ uint256 half up
       ✓ int256 half up
       ✓ int256 ceil
       ✓ int256 down
    others
       ✓ uint256 max
       ✓ uint256 min
       ✓ int256 max
       ✓ int256 min
```

```
    ✓ abs
    ✓ neg

Liquidate
  basic
    ✓ liquidateByAMM (163ms)
    ✓ liquidateByAMM - bankrupt (117ms)
    ✓ liquidateByAMM - vault fee (162ms)
    ✓ liquidateByAMM - vault fee / bankrupt (122ms)
    ✓ liquidateByAMM - vault fee / not bankrupt (134ms)

LiquidityPool
  2 liquidityPool group
    ✓ updatePrice (127ms)
    ✓ getAvailablePoolCash (127ms)
    ✓ rebalance (248ms)
    ✓ rebalance - 2 (160ms)
    ✓ isAMMMaintenanceMarginSafe (184ms)
  operator
    ✓ transferOperator (118ms)
  trader
    ✓ donateInsuranceFund (78ms)
    ✓ deposit (53ms)
    ✓ withdraw (87ms)
    ✓ clear-no-account (45ms)
    ✓ clear (257ms)
    ✓ clear - 2 (335ms)
    ✓ clear - 3 (306ms)
    ✓ settle (324ms)
    ✓ updateInsuranceFund (74ms)

LiquidityPool2
  liquidity
    ✓ init (147ms)
    ✓ before safe, after safe (150ms)
    ✓ short, before unsafe, after unsafe (159ms)
    ✓ short, before unsafe, after safe (151ms)
    ✓ long, before unsafe, after unsafe (149ms)
    ✓ long, before unsafe, after safe (152ms)
    ✓ invalid margin to add (96ms)
    ✓ poolMargin = 0 && totalShare != 0 (107ms)
    ✓ donate (104ms)
  remove liquidity
    ✓ no position (191ms)
    ✓ no position, remove all (191ms)
    ✓ short (197ms)
    ✓ long (203ms)
    ✓ state != NORMAL (193ms)
    ✓ all states CLEARED  (189ms)
    ✓ no position (200ms)
    ✓ no position, remove all (202ms)
    ✓ short (203ms)
    ✓ long (205ms)
    ✓ state != NORMAL (189ms)
    ✓ all states CLEARED  (194ms)
    ✓ zero share to remove (151ms)
    ✓ short, before unsafe (166ms)
    ✓ long, before unsafe (160ms)
    ✓ short, after unsafe (171ms)
    ✓ long, after unsafe (170ms)
    ✓ long, after negative price (177ms)
    ✓ long, after exceed leverage (176ms)
    ✓ pool margin = 0 (147ms)
    ✓ zero index (93ms)
    ✓ zero supply of share token (137ms)

  LiquidityPool3
    ✓ createPerpetual - address (103ms)
    ✓ createPerpetual - fastCreation disable (170ms)
    ✓ createPerpetual - fastCreation enabled (239ms)

  LoopTest
200000000000000000
share: 0.0
ctk  : 10001.652670961063243546
┌─────────┬──────────────────────┬──────────────┐
│ (index) │          0           │      1       │
├─────────┼──────────────────────┼──────────────┤
│    0    │ 'deposit           ' │ '   336,006' │
│    1    │ 'addLiquidity      ' │ ' 3,324,522' │
│    2    │ 'trade 1 - open    ' │ ' 3,779,930' │
│    3    │ 'trade 2 - open    ' │ ' 3,410,799' │
│    4    │ 'trade 3 - revert  ' │ ' 3,360,433' │
│    5    │ 'trade 4 - close all' │ ' 3,280,451' │
│    6    │ 'withdraw          ' │ ' 2,900,275' │
│    7    │ 'removeLiquidity   ' │ ' 3,352,128' │
└─────────┴──────────────────────┴──────────────┘
    ✓ main (8610ms)
┌─────────┬──────────────────────┬──────────────┐
│ (index) │          0           │      1       │
├─────────┼──────────────────────┼──────────────┤
│    0    │ 'deposit           ' │ '   336,006' │
│    1    │ 'addLiquidity      ' │ ' 3,324,522' │
│    2    │ 'trade 1 - open    ' │ ' 3,779,930' │
│    3    │ 'setEmergencyState ' │ ' 5,506,133' │
└─────────┴──────────────────────┴──────────────┘
    ✓ settle (5636ms)

  LpGovernor
    ✓ mint / redeem (79ms)

  MarginModule
    Getters
      ✓ +getInitialMargin (47ms)
      ✓ -getInitialMargin (56ms)
      ✓ +getMaintenanceMargin (50ms)
      ✓ -getMaintenanceMargin (39ms)
      ✓ +margin (47ms)
      ✓ -margin (47ms)
      ✓ +position (39ms)
      ✓ -position (44ms)
      ✓ getAvailableCash + funding (39ms)
      ✓ getAvailableCash - funding 1 (45ms)
      ✓ getAvailableCash - funding 2
      ✓ +isInitialMarginSafe yes (54ms)
      ✓ -isInitialMarginSafe yes (42ms)
      ✓ +isInitialMarginSafe no (45ms)
      ✓ -isInitialMarginSafe no (48ms)
      ✓ +isMaintenanceMarginSafe yes (54ms)
      ✓ -isMaintenanceMarginSafe yes (48ms)
      ✓ +isMaintenanceMarginSafe no (46ms)
      ✓ -isMaintenanceMarginSafe no (44ms)
      ✓ isEmptyAccount (46ms)
      ✓ isEmptyAccount - 1 (43ms)
      ✓ isEmptyAccount - 2 (50ms)
    Setters
      ✓ setMarginAccount (101ms)
    OpenInterest
      3) "before all" hook for "updateMargin"

  Minging
    ✓ notifyRewardAmount (262ms)
    ✓ setRewardRate (76ms)
    ✓ earned (287ms)
    ✓ rewardPerToken (232ms)
    ✓ rewardPerToken - 2 (258ms)
    ✓ rewardPerToken - reward tuncation (499ms)

  OracleRouter
    ✓ hash
    ✓ normal
    ✓ closed / terminated (64ms)

  Order
    ✓ signature
    ✓ validateOrder (64ms)
0xc6dd6530bc669ead14b253033fd6267e180b98cecf9d67a0df3955472552e867
    ✓ decompress
orderHash: 0xc6dd6530bc669ead14b253033fd6267e180b98cecf9d67a0df3955472552e867
signature: 0x306af4b96ab19e1f178349ee072b7777e74655c618fea94bcc5fea4a488a45094c727203780b0c635cbc9c772a85071df215e90c635b72642ca321a2061230231b
    ✓ signer

  Perpetual
    ✓ getMarkPrice && getIndexPrice (62ms)
    ✓ getRebalanceMargin (105ms)
    ✓ setNormalState (71ms)
    ✓ setEmergencyState (71ms)
    ✓ setClearedState (70ms)
    ✓ deposit (49ms)
    ✓ withdraw (98ms)
    ✓ withdraw - market closed (81ms)
    ✓ clear (152ms)
    ✓ clear - 2 (237ms)
    ✓ getNextActiveAccount (109ms)
    ✓ settle (261ms)
```

```
Perpetual2
  erc20
     ✓ deposit (114ms)
     ✓ withdraw (190ms)
     ✓ withdraw - wrapped (197ms)
     ✓ trade - 1 (248ms)
     ✓ trade - 2 (354ms)
     ✓ settle (349ms)

Reader
   ✓ getAccountStorage
   ✓ getLiquidityPoolStorage (94ms)
   ✓ zero price (112ms)

RemarginHelper
   ✓ main (964ms)

SymbolService
  whitelisted factory
     ✓ add and remove
     ✓ not owner
  allocate symbol
     ✓ normal (64ms)
     ✓ not contract
     ✓ wrong factory
     ✓ perpetual exists
     ✓ not enough symbol (100ms)
  assign reserved symbol
     ✓ normal (41ms)
     ✓ not owner
     ✓ not contract
     ✓ wrong factory
     ✓ symbol too large
     ✓ symbol exists (54ms)
     ✓ invalid symbol (52ms)

TradeModule1
  basic
     ✓ getFees (79ms)
     ✓ getFees - rebate (132ms)
     ✓ getFees - open (70ms)
     ✓ validatePrice
    postTrade
       ✓ hasOpenedPosition (54ms)
       ✓ postTrade - 1 (73ms)
       ✓ postTrade - 2 (59ms)
       ✓ postTrade - 3 (66ms)
    trade
       ✓ sell (128ms)
       ✓ buy without cross 0 (117ms)
       ✓ buy cross 0 (148ms)

  TradeModule2
    basic
       ✓ broker (180ms)
7698.86 2.3
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
0x70997970C51812dc3A010C7d01b50e0d17dc79C8
       ✓ broker - 2 (77ms)
0x46dd8af0f79c228b984bfdcf863e5153e223dc5a11e1882da9a3213647f45da9
0x4ea20698b634a5cf2f486624886b98beb839f1de727d80cfa99288cf9962a08a
28
orderHash 0xa1d61d5001a81fd4ba8ae4a25f7dc0d0e18a092525f602c16bb2645ab1aaffc4
       ✓ test

  TradeModule3
    basic
       ✓ regular (150ms)
       ✓ close (268ms)
       ✓ close - but no fee (425ms)
       ✓ market (174ms)

  TradeModule4 - auto deposit/withdraw with targetLeverage
    basic
       ✓ regular - 1x (177ms)
       ✓ close (193ms)

  upgrade
     ✓ main (1427ms)
     ✓ main - 2 (1468ms)

  normal
     ✓ 1 oracle, vanilla (69ms)
     ✓ 1 oracle, inverse (74ms)
     ✓ 2 oracles, vanilla (82ms)
     ✓ 2 oracles, inverse (79ms)
     ✓ 3 oracles (91ms)


  345 passing (3m)
  3 failing

  1) Creator
       versionControl:
     AssertionError: Expected transaction to be reverted with implementation is already existed, but other exception was thrown: Error: VM Exception while processing transaction: reverted with reason string 'implementation
already exists'


  2) Governance
       "before each" hook for "checkIn":
     Error: VM Exception while processing transaction: reverted with reason string 'invalid oracle address'
      at PerpetualModule.setClearedState (contracts/module/PerpetualModule.sol:389)
      at PerpetualModule.initialize (contracts/module/PerpetualModule.sol:147)
      at TestGovernance.initializeParameters (contracts/test/TestGovernance.sol:43)
      at runMicrotasks (<anonymous>)
      at processTicksAndRejections (internal/process/task_queues.js:93:5)
      at runNextTicks (internal/process/task_queues.js:62:3)
      at listOnTimeout (internal/timers.js:523:9)
      at processTimers (internal/timers.js:497:7)
      at HardhatNode._mineBlockWithPendingTxs (node_modules/hardhat/src/internal/hardhat-network/provider/node.ts:1261:23)
      at HardhatNode.mineBlock (node_modules/hardhat/src/internal/hardhat-network/provider/node.ts:384:16)

  3) MarginModule
       OpenInterest
         "before all" hook for "updateMargin":
     Error: VM Exception while processing transaction: reverted with reason string 'oracle must be contract'
      at PerpetualModule.setClearedState (contracts/module/PerpetualModule.sol:389)
      at PerpetualModule.initialize (contracts/module/PerpetualModule.sol:147)
      at TestMarginAccount.createPerpetual (contracts/test/TestPerpetual.sol:30)
      at runMicrotasks (<anonymous>)
      at processTicksAndRejections (internal/process/task_queues.js:93:5)
      at runNextTicks (internal/process/task_queues.js:62:3)
      at listOnTimeout (internal/timers.js:523:9)
      at processTimers (internal/timers.js:497:7)
      at HardhatNode._mineBlockWithPendingTxs (node_modules/hardhat/src/internal/hardhat-network/provider/node.ts:1261:23)
      at HardhatNode.mineBlock (node_modules/hardhat/src/internal/hardhat-network/provider/node.ts:384:16)
```

Code Coverage

Some critical modules have not enough coverage above 80%. We recommend fixing the 2 failing tests and check again if Cove Coverage improves. Otherwise, add more tests, especially for `LiquidityPool`.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	64.9	47.27	60.87	66.03	
Getter.sol	55.36	0	50	57.41	… 395,399,401
Governance.sol	18.75	16.67	14.29	20.59	… 159,160,161
LibraryEvents.sol	100	100	100	100	
LiquidityPool.sol	86.67	68.75	87.5	81.25	29,160,161
Perpetual.sol	100	63.46	100	100	
Storage.sol	100	66.67	100	100	
Type.sol	100	100	100	100	
contracts/broker/	83.61	75	90	82.26	
Broker.sol	83.61	75	90	82.26	… 189,191,192
contracts/factory/	87.31	61.43	84.78	87.5	
AccessControl.sol	94.12	91.67	100	94.12	67
PoolCreator.sol	100	62.5	100	100	
Tracer.sol	81.25	58.33	85.71	81.82	… 114,140,181
Variables.sol	75	35.71	77.78	75	43,80,81,83,84
VersionControl.sol	82.14	62.5	72.73	82.14	… 213,214,215
contracts/libraries/	82.82	77.94	85.71	83.23	
BitwiseMath.sol	100	100	100	100	
Constant.sol	100	100	100	100	
EnumerableMapExt.sol	46.15	37.5	57.14	44.44	… 124,126,127
Math.sol	100	100	100	100	
OrderData.sol	91.67	50	83.33	92.31	94,103
SafeMathExt.sol	100	100	100	100	
Signature.sol	77.78	25	100	88.89	37
Utils.sol	66.67	68.75	71.43	66.67	… 125,126,128
Validator.sol	100	100	100	100	
contracts/module/	94.08	75	90.23	93.89	
AMMModule.sol	99.46	91.84	100	99.47	517
CollateralModule.sol	88	50	80	88	67,89,104
LiquidityPoolModule.sol	89.63	69.33	80	89.11	… 94,995,1041
MarginAccountModule.sol	100	90	100	100	
OrderModule.sol	80.95	67.65	100	80	43,53,122,128
PerpetualModule.sol	92.68	66.1	93.33	92.64	… 281,645,646
TradeModule.sol	100	86.21	100	100	
contracts/oracle/router/	100	85	100	100	
OracleRouter.sol	100	87.5	100	100	
OracleRouterCreator.sol	100	75	100	100	
contracts/oracle/simple/	77.42	75	92.86	77.42	
OracleWrapper.sol	77.42	75	92.86	77.42	… 56,57,58,59
contracts/remargin/	80	37.5	100	80	
RemarginHelper.sol	80	37.5	100	80	20,21
contracts/symbolService/	100	95.83	100	100	
SymbolService.sol	100	95.83	100	100	
All files	**88.8**	**70.81**	**85.4**	**88.75**	

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

```
ffc4cb8761a5674b775fc363852956080d048ba8dfc37484c8f874bb4914cae8 ./contracts/Type.sol
2940dc69bc49b2dde34e00206428aaf27ef02cb4e9a7e50b4d77b7896d4cc56b ./contracts/Governance.sol
04e91bb977148e8b54a6bb527fc75f23684d09304dfb10432129f21dfa9a22eb ./contracts/Getter.sol
493373330cb4e84889d4e150a9d50e2ec5868545da41f2d4c27c6de0753c7a87 ./contracts/Perpetual.sol
50b4852ee96d8ac501c486c42243dc40b19ba604138c988141d7537f1748d90e ./contracts/Storage.sol
41deaa42f4b87779e009bf9d513758c87a6f31a423f4b4e3efd662a54681a95d ./contracts/LibraryEvents.sol
ceb761a46971e219b599ee4358698bc01d0584ddf64bffb246ddee4e0f55b9f6 ./contracts/LiquidityPool.sol
afcf7265bebafbcabe8dd98de470ed3920db9d93127cdbf7b3df5e91f7729813 ./contracts/factory/PoolCreator.sol
1692c2acd83ffa6483a48429aff0367dfcb5fa659962f79f90584126deca5816 ./contracts/factory/Tracer.sol
3c32fc987f5dfed100b8d95bcf27f582c56f3d8e2b0d38dcfdd0233f408ba54f ./contracts/factory/Variables.sol
76a9411cff1f0b87079df626c918f4de1695d499077513881747f0802bb93725 ./contracts/factory/VersionControl.sol
0f9b66f2dcfe97e06bf8eefb17c718dba6fb97e1a2407edc19cdbd25a74b1a22 ./contracts/factory/AccessControl.sol
587d3561da6b04e4f8aa86833ca75499d4cd65dbdd528e7f713b5f46a3b208d8 ./contracts/governance/RewardDistribution.sol
67b58526e3c30d6f40b793bd041e23ac325603a0cd2d6c92b104927dc9142601 ./contracts/governance/GovernorAlpha.sol
e4deeacf60f69ca0d302097980d753b7081677e5656227ff2098868a8b43b1a4 ./contracts/governance/LpGovernor.sol
026c2305d26871df758edb25c134ec52188f23cf742314b83cac16d3338eabde ./contracts/libraries/BitwiseMath.sol
7667c438dc72c1fb7676b014fc54da1de705c4d520be516c849f106de126eef7 ./contracts/libraries/Constant.sol
48023f329051ac878df34c46441ac1f3b93782b92013bd233f629f7755672768 ./contracts/libraries/Validator.sol
5b9a8bc82767420f52642a7c9cc47afdceb8db555b2da39531a467559594aa92 ./contracts/libraries/OrderData.sol
49ba9d35990e008da327e76b9f672ff921499451fdd5909d3aa8981a0d15e326 ./contracts/libraries/EnumerableMapExt.sol
28fc7c9084b1a35502af5d02d42df84f25d76397753f939bcf1645ee001d6cca ./contracts/libraries/SafeMathExt.sol
07cde1b1e0ecef20886988953a476f6cc4a51574bca73e09eb0215fe310d8854 ./contracts/libraries/Math.sol
acffe150e006df701fc986a324135e806650825e1fbc15f3bb47c3fb051a3729 ./contracts/libraries/Signature.sol
c1a12e11a3bc9720d8cab7261fc055ef0a0c837299f3c14512097a3558dd0cfe ./contracts/libraries/Utils.sol
96dd6ccd75d91182507d8137acd88f9e2649a73d69810e4ee5555a730a15173d ./contracts/oracle/router/OracleRouter.sol
1d8368419e5fc39a85425b33df5372b047d103bd104a6e0ea446ba80c4293f0c ./contracts/oracle/router/OracleRouterCreator.sol
6d525cbdee2f8956302eeade2a0260f9295eda1d10c0ad3a38a0acc9a44eb7dd ./contracts/oracle/simple/OracleWrapper.sol
140ae2e11151ac5a7f3656160691a8d87b9a7767fe4466f9e23cbccd69f76b01 ./contracts/remargin/RemarginHelper.sol
8a9a0bd59a384e4ddf260dfae4fc409957554740526f0a48235a5e70422d4014 ./contracts/module/PerpetualModule.sol
547cc20f3d8997d8ac35f955e5423f84af58f43d4d27badd776bd8443d384976 ./contracts/module/TradeModule.sol
1bfddc695e8a4baa2e3604f2d4361d94c6a9c3c2949a7748bd3c53f573d404f6 ./contracts/module/OrderModule.sol
5ec0bc5367b6c45f87356b17b96155221f7bd056187a31c24476bdbadc66eb2e ./contracts/module/CollateralModule.sol
a2c10dde16a1cf6937ef4a9c635035ff4662d6af3264c13ac374309eab761f63 ./contracts/module/LiquidityPoolModule.sol
1260de9ac984edccf84e0b613787f2f69617f58c7893e68896878670fe556756 ./contracts/module/AMMModule.sol
dbd53887aa0d900ebe125ad9a92253fb5d47772797456524a1e6f444c4629b88 ./contracts/module/MarginAccountModule.sol
1493a543f8f978acc9b239dd68262a425bd17ed701e8a0765db828846f89ab15 ./contracts/test/TestAccessControl.sol
205242f05217bb930fda600961240daf7cee75f96b69b04c57fe76b50ce4f20d ./contracts/test/TestLiquidityPool.sol
6f9e964687cba1b7cc9bb6b325e46d89093f2699f659390effc069a14571277d ./contracts/test/WETH9.sol
f10bc0b112c2d9945f1976911839612e8a7b32b71657db9afd05e070fb1a2f44 ./contracts/test/MockPoolCreator.sol
0f0b11ad2b6e6d3ee8f4acb19ed5b43570b6d5bd08add729256bd938c81e0ada ./contracts/test/TestTrade.sol
3072d001798d5af9d3ee407adfd77950d906ca36d28ec4ab795ffd76b839cc9b ./contracts/test/TestGovernor.sol
ab352560989673c8494718352c8b00d01e69f8068bd3fc6491d01d6606a28426 ./contracts/test/MockAMMModule.sol
dfae47c5e58fc70062892d7f861c269ac95d9f8a263fc5b30b7aec9b73e4a5f5 ./contracts/test/TestSymbolService.sol
c4aea62ddb87bd325ba157d3a7bc765a8b5fa71896c09b44b16e895c5771f145 ./contracts/test/TestTracer.sol
22ae833f11ab2a2e331edd271e9845e165aa43ff7f03c02c23fff2a173c9daeb ./contracts/test/TestBitwiseMath.sol
75ef67cba6c9f377f4d4569fadcc3c7f4462550da69382fa2fafb032822c673e ./contracts/test/BlackHole.sol
a300c74057ce1ca519d87a6b4cd42744061b0a2ff65565133cef5aafedc91ec2 ./contracts/test/TestLibSafeMathExt.sol
688a695142f5a43c2305d1dfe5dd6507bc153b0ef8b82dfd80308a0bf09c81f3 ./contracts/test/MockLiquidityPool.sol
955ab27daf3227b6dcb7c976d3e47ecee5c4cb477bbb51c0b1417cb6b13ee811 ./contracts/test/CustomERC20.sol
f3b86452921bd9642d9f72ea809661e2aa946126e9d1e15de84a9a18efee6aae ./contracts/test/TestPerpetual.sol
bafc562f86f7bf84c2102c1b18ade56e643db7e8789067ea18384873c676a302 ./contracts/test/TestShareToken.sol
8c967278f71d41e234d304755e2995b21ddb5a9eb9cf654451c45f193e1cdff5 ./contracts/test/TestCalc.sol
192605ada177d1ca44ecc9942d0c11fcb4e7912d3455bc01bfc59d108f906863 ./contracts/test/TestLiquidityPoolUpgraded.sol
f113621580a0704a0423f194155a52ad54d347257829a981ad28d36c37d38354 ./contracts/test/TestMarginAccount.sol
```

0aa6fe110fcaf66df6f4343f13536dbe5076dc6ad0d065cd0ef69c91f762e078 ./contracts/test/TestGovernance.sol

05671f976d81bb1b83d9a7b7b91a75a2ae93c9d4b713070be3b80559a925a0ca ./contracts/test/TestAMM.sol

245ed522214336e6fa3363de605511a9aa456dd5b8aa0c3594ce37fd99e98785 ./contracts/test/TestLpGovernor.sol

5d9299fdc8aa00a1a6e8347d0ff9ec5f36b20d13a7e7a426f38055cad9378b95 ./contracts/test/LiquidityPoolAdmin.sol

90102616f2bae0bc23dc2102049f75df7331d0abe7e4aa9c21b1259f9035edf4 ./contracts/test/TestLpGovernorUpgraded.sol

89e5bbddd7c802552cf3f50bf252721c6f0a6b2b731905c63d778c998c53b3d1 ./contracts/test/TestHelper.sol

e9da335b66a0227d2dde15f89980fa14ade34fbaac398a7769358d421337ca09 ./contracts/test/TestOrder.sol

a1205887ae50e80bcdf7ce912e35a0151776ecc6cfda3d82a6b93c619906a050 ./contracts/test/TestLibMath.sol

c6c4f106d478c9d0639406cee10a29e72a724b7e34926663279cb21ab7328199 ./contracts/broker/Broker.sol

8d42f3544774fd93a6203ff9a714aa5c76f55ce682dc232f7ba790b56135b0f9 ./contracts/symbolService/SymbolService.sol

fc496929b24f92acba7bd1b4cf81331ba61f96bea47d2201a0efb5011de9edaa ./contracts/interface/IPerpetual.sol

671ae00789c199971b4ea32111279b223568b26a96518ead1d31e5bf58bc6654 ./contracts/interface/ITracer.sol

a37d6382d6f7c9d601e1fef1a3509c89fef2b721e2d3411510fc0e9202a4ba08 ./contracts/interface/IVariables.sol

7c471a5cb5b7c13a833263e331f8ff0e92b0a3b8f1a116d7f564e27cb817cfe9 ./contracts/interface/ILiquidityPoolGetter.sol

069d65e372e267d99f01e85e7d5c86d06f49db7214a90a970493300d51e59161 ./contracts/interface/IPoolCreator.sol

013ebfbe63674438c24d06c6c05114b4eed916e8acfe2f40f9f2fb2da3bb9f45 ./contracts/interface/IDecimals.sol

82ade868fe6ef6d298c0b03e2d57dce0ba0c3497c8d50ce88bc2a84673582dc9 ./contracts/interface/IPoolCreatorFull.sol

07f7adb8db8208cc498349abf950962e3dc0041cf5304d78ab62245369f1c900 ./contracts/interface/IOracle.sol

2c3d147abbd39b844e9be0eaec6835af454fcd276a88266c8866ad022b45a28b ./contracts/interface/IVersionControl.sol

84d9fc449e580fdd15b35360a0a44268ed2d35394500387ba964e8c2a8c9c3af ./contracts/interface/IAccessControl.sol

4d4d268c2e92b2d515eed31ba200bd70b54a208564b414adeaf1f432f47b4e64 ./contracts/interface/ILiquidityPoolGovernance.sol

cf1d470332cc342590624f8dc36bcd9e62c70cf0045e57142e80f24bc2a8ba90 ./contracts/interface/ILiquidityPool.sol

490b52448d2cc6dd435ec787cd882d8d1c3cf7107648ff979d10552604e1b050 ./contracts/interface/IGovernor.sol

1bad23386096807a8f9dc610c48634a35bb45bb6df562f20cdacb3cc67d9a7f8 ./contracts/interface/ILiquidityPoolFull.sol

b4b3c0bd8b3de4b3f4fa083a99b3d58a607f5d775b9297b9ada62e7015767329 ./contracts/interface/ISymbolService.sol

0b40f4c644cd83a9fd9b5116a399827826fdac53a7693b69e7ed363226669c9b ./contracts/interface/IProxyAdmin.sol

986ad256318ea0b5a806f0e35090bd116c7f47094a5f1c273392ddd18ab630cf ./contracts/interface/IUpgradeableProxy.sol

f0754f62ab9e3db6f6cd86ce17f318c55909bebcbf0de32d3b70dca036053c5d ./contracts/reader/Reader.sol

Tests

93e9db92ffff47d67492bad7fb52eb5cb3a4b9ebbd3238475bd1843cf90b5144 ./test/Integration3.test.ts

2912e19185a9fd973886ff6a96425568189b00f1bb5b9def0ea8a9a745886624 ./test/Funding.test.ts

d7af90f03593f498a5849f66957336e6514bfa8cb35b00bd812d518e1a314f3b ./test/LiquidityPool3.test.ts

f01d7fa5050f9e170e3ad9b4d11a4540af916278b53a8bb9358442b8ae2e19e8 ./test/SymbolService.test.ts

89dfcbcf9aa36ad9c595d76d6dafc28a69a5b7561e33abfb57d730427db3da31 ./test/OracleRouter.test.ts

cfe4c9d709f68755427d75571d07d3461ca813242f887e82dbb1bae94e0b59c0 ./test/Integration2.test.ts

7ceaf2b5cecd89cb49a9bf3da969613cd837c969948f174ca2d12e6a13de9606 ./test/LiquidityPool2.test.ts

3a185d9e3a315f4f28daefde3d9d30d5185a236a24a62be1b8ebc835aac98b30 ./test/MarginAccount.test.ts

4f857ce78ae84ac5880a8e3f80c2daa2d21a8ff9861bba58cae32cd7089b184a ./test/LiquidityPool.test.ts

616cce5677adc8a566b126e4e5ec06d61461d80282edb09ab73df7ce27171b09 ./test/BitwiseMath.test.ts

e022a1e6e86cedec3e4242470e088c60cd4bb2774e197957164a0f9b081d8e5c ./test/helper.ts

e9ad5315f525b52aba031bc6fe89b424520b5ecdb09d6ab4b129f9381c5de378 ./test/Trade4.test.ts

b9a7ed6f4f266910e74a091e8ecfda0998d3106bfc48241d6fda309a982e4194 ./test/Order.test.ts

223916251698a0287cf7ac8e77e7b902f2c714c7a706f8f3a333ad8921f5bd7c ./test/LoopTest.test.ts

589e1e5bd70034cda794b63f1ddd6a753e4ffcc8d8e5c75a1e75c51dc14d1663 ./test/Perpetual.test.ts

b9a75c99f969a48d52e06ef049a6761095a02f18dc01bfff449a1df940ab95c8 ./test/RemarginHelper.test.ts

776d2c5664ff25aa675b3b74c477cf632f93577183d7c5420027390ba7beada8 ./test/GovernorAlpha.test.ts

eebec8bec6732cee883d8495ac1aa0aa67ef836d2e4c76aefa4973c605b38d80 ./test/Upgrade.test.ts

83b0963598ca37777dd053893ee4b10562d9a071cdc93efa60ecee4bd9d92d7a ./test/Trade1.test.ts

ac4e9abed731cecfba074f87bb90e2879502ae2c9326049aaa082d63cb480679 ./test/LibMath.test.ts

7e184c7ef80d4531faeb631b87de987a0a94eccb56834686e1fbfd8d35da91cb ./test/AMM.test.ts

64563f938269ae519e8ae9c389b821e527243f8c4b34ede967018c066af62453 ./test/Reader.test.ts

01bda26da2034f49bd02b981b209a5dbfa34cdd19a3e8fdb45e19b4e0db21e95 ./test/Perpetual2.test.ts

12b1f359508e52bfcbf8baef07980d07eff46941e91c40e0cbc0d967f96e2e22 ./test/AccessControl.test.ts

45ad0bc894568325ef2b237c6b0869e7f79c8d99be08c15872c5f5e1e59fc907 ./test/Trade2.test.ts

a100b06b3b7e444db34ad85a78b758146880dbc33744bf0dbf840a23e4a89208 ./test/Governance.test.ts

b6b1b3b9b95f5d3a557d887056818964fd5688ff41b653090e0b04fb82a56c37 ./test/LpGovernor.test.ts

e7f47e3777040edd1077d6c60f3568b01f40dbeb550f5da99b225a887363e5d2 ./test/Mining.test.ts

f55162db8a4c9b39117915ce93fc06faad32baa6a8e0c08e3c8d8dc3e376ea10 ./test/Integration.test.ts

06918d36875138369617fd7476f4d36131c54e46d6df4c2533642bf0d973edd4 ./test/Integration2Lev.test.ts

97bb0f310456dcc4793edd268a2f9345b1d5b97c1ba4661708fdd0883e8c53e4 ./test/LibSafeMathExt.test.ts

4ed2ffbeb46aea89e49cc107a3b022d5b1448e999d0d03e107d40517161b6d3a ./test/Creator.test.ts

23bbb283edba36e8aebc80126f05804d1ae223cb7e90bcd19df06f84d47ef0f8 ./test/Getter.test.ts

e9305338fcf7389b77f8a3df0cc8bbe570de8ba77cede5fdc278d25a22343ec3 ./test/Broker.test.ts

e63a51c6e6d6ed3229758011d29b48c724eb44162f7ba412722f7712f76eb0ed ./test/Trade3.test.ts

907838ad7ac5416f03ec14487a5cf137d38f8372f6c7ee3cb31eea7d8b56141f ./test/Liquidate.test.ts

Changelog

- 2021-06-21 - Initial report [5c67638]
- 2021-07-06 - Reaudit report ['edf2053']

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.