Quantstamp Security Assessment Certificate

October 14th 2021 — Quantstamp Verified

Fodl

This audit report was prepared by Quantstamp, the leader in blockchain security.

Executive Summary

Type	DeFi protocol
Auditors	Kacper Bąk, Senior Research Engineer Ed Zulkoski, Senior Security Engineer Hisham Galal, Research Engineer
Timeline	2021-09-01 through 2021-10-13
EVM	London
Languages	Solidity, TypeScript
Methods	Architecture Review, Unit Testing, Computer-Aided Verification, Manual Review.
Specification	Fodl
Documentation Quality	Medium
Test Quality	High

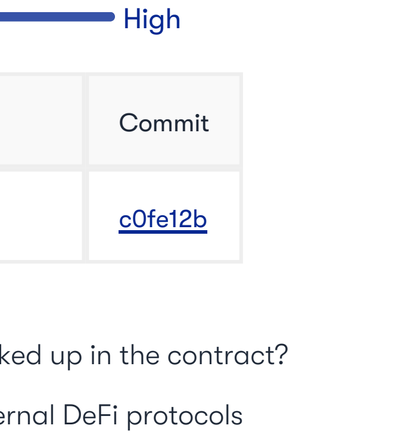
Source Code

Repository	Commit
fodl	c0fe12b

Goals

- Can funds get locked up in the contract?
- Does Fodl use external DeFi protocols correctly?
- Are calculations implemented correctly?
- Are taxes collected as expected and they cannot be circumvented?
- Are business requirements reflected in the implemented logic?
- Is the diamond pattern implemented correctly?

Total Issues	**11** (6 Resolved)
High Risk Issues	0 (0 Resolved)
Medium Risk Issues	0 (0 Resolved)
Low Risk Issues	**4** (2 Resolved)
Informational Risk Issues	**6** (3 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

After reviewing the codebase, we have found a number of low-severity and informational issues. Overall, the code appears to be well-written and properly tested. It is lacking, however, good documentation. We recommend improving the documentation and addressing the indicated issues.
Update: the team addressed the issues we indicated.

ID	Description	Severity	Status
QSP-1	Approvals are not reset to 0	Low	Acknowledged
QSP-2	`createAccount()` may fail if existing NFT is transferred	Low	Fixed
QSP-3	Uniswap exchanger does not allow multi-step paths	Low	Acknowledged
QSP-4	Unchecked function arguments	Low	Fixed
QSP-5	Allowance Double-Spend Exploit	Informational	Mitigated
QSP-6	Interaction with External Contracts	Informational	Acknowledged
QSP-7	Privileged Roles and Ownership	Informational	Acknowledged
QSP-8	Inaccurate `BLOCKS_PER_YEAR`	Informational	Acknowledged
QSP-9	Constraints do not match inline comments	Informational	Fixed
QSP-10	Clone-and-Own	Informational	Fixed
QSP-11	Unclear use of `create2` opcode	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 v0.8.0

Steps taken to run the tools:

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

Findings

QSP-1 Approvals are not reset to 0

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `Fodl/modules/Exchanger/UniswapExchangerAdapter.sol`

Description: Approvals are not reset to 0 after swaps in `exchange()`, `swapFromExact()` and `swapToExact()`.

Recommendation: Reset approvals to 0.

Update: The team explained that for now, these are designed to interact with Uniswap V2 and Sushiswap, which they consider time tested protocols they can trust. They will consider adding it if they interact with a UniswapV2 fork with less reputation.

QSP-2 `createAccount()` may fail if existing NFT is transferred

Severity: *Low Risk*

Status: Fixed

File(s) affected: `Fodl/core/FoldingRegistry.sol`

Description: When creating a new account, the `create2` salt is defined as `uint256 salt = uint256(keccak256(abi.encodePacked(msg.sender, fodlNFT.balanceOf(msg.sender))));`.
Since the NFT is an ERC721, it may be transferred to a different address. Suppose the user has created one account, but then transfers it to a different address (such that `fodlNFT.balanceOf(user)` is reduced back to 0). On a second invocation of `createAccount()`, the salt `uint256(keccak256(abi.encodePacked(user, 0)))` will be exactly the same as the first invocation, and the `create2` call will fail.

Recommendation: Consider a per-user nonce variable that increments for each `createAccount()` call, but is separate from the user's NFT balance.

Update: `create2` is used to allow for deterministic creation of contract accounts. This is needed in order to pre-approve tokens expenditure from the account owner. It is done this way so that a create account call and an open position call can be done at the same time in the FoldingRegistry, so that UX is smoother. They added private nonces to the registry to address this issue.

QSP-3 Uniswap exchanger does not allow multi-step paths

Severity: *Low Risk*

Status: Acknowledged

File(s) affected: `Fodl/modules/Exchanger/SushiswapExchangerAdapter.sol`

Description: The function `exchange()` constructs a routing path between the `fromToken` and the `toToken`, however not all token pairs have a direct routing path in Uniswap. In such cases, it is often sufficient to create a routing path that first converts `fromToken` to ETH, and then from ETH to the `toToken`.
The same issue exists for all functions that construct a routing path in the contract.

Recommendation: Consider allowing routing paths larger than length 2.

QSP-4 Unchecked function arguments

Severity: *Low Risk*

Status: Fixed

File(s) affected: `Fodl/modules/FundsManager/FundsManager.sol`, `Fodl/core/FoldingAccount.sol`, `Fodl/core/FoldingRegistry.sol`

Description:

1. `FoldingRegistry.initialize()` does not check that each address argument is non-zero. **Update: ** fixed.
2. `FoldingRegistry. addImplementation()` does not check that each address argument is non-zero. **Update: ** fixed.
3. `FoldingRegistry. addPlatformWithAdapter()` does not check that each address argument is non-zero. **Update: ** fixed.
4. `FoldingRegistry. changePlatformAdapter()` does not check that each address argument is non-zero. **Update: ** fixed.
5. `FoldingRegistry. addCTokenOnPlatform()` does not check that each address argument is non-zero. **Update: ** fixed.
6. `FoldingRegistry. addExchangerWithAdapter()` does not check that each address argument is non-zero. **Update: ** fixed.
7. `FoldingRegistry. changeExchangerAdapter()` does not check that each address argument is non-zero. **Update: ** fixed.
8. `FoldingAccount.constructor()` does not check that each address argument is non-zero. **Update: ** acknowledged.
9. `FundsManager.constructor()` does not check that each address argument is non-zero. It also does not check that `_profit` and `_principle` are both `<= MANTISSA`. **Update: ** fixed.

Recommendation: Add `require` statements to each function as suggested.

QSP-5 Allowance Double-Spend Exploit

Severity: *Informational*

Status: Mitigated

File(s) affected: `FodlToken/FodlToken.sol`

Description: As it presently is constructed, the contract is vulnerable to the allowance double-spend exploit, as with other ERC20 tokens.

Exploit Scenario:

1. Alice allows Bob to transfer `N` amount of Alice's tokens (`N>0`) by calling the `approve()` method on `Token` smart contract (passing Bob's address and `N` as method arguments)
2. After some time, Alice decides to change from `N` to `M` (`M>0`) the number of Alice's tokens Bob is allowed to transfer, so she calls the `approve()` method again, this time passing Bob's address and `M` as method arguments
3. Bob notices Alice's second transaction before it was mined and quickly sends another transaction that calls the `transferFrom()` method to transfer `N` Alice's tokens somewhere
4. If Bob's transaction will be executed before Alice's transaction, then Bob will successfully transfer `N` Alice's tokens and will gain an ability to transfer another `M` tokens
5. Before Alice notices any irregularities, Bob calls `transferFrom()` method again, this time to transfer `M` Alice's tokens.

Recommendation: The exploit (as described above) is mitigated through use of functions that increase/decrease the allowance relative to its current value, such as `increaseAllowance()` and `decreaseAllowance()`.

Pending community agreement on an ERC standard that would protect against this exploit, we recommend that developers of applications dependent on `approve()` / `transferFrom()` should keep in mind that they have to set allowance to 0 first and verify if it was used before setting the new value. Teams who decide to wait for such a standard should make these recommendations to app developers who work with their token contract.

QSP-6 Interaction with External Contracts

Severity: Informational

Status: Acknowledged

Description: The protocol relies on functionalities of external contracts. Therefore, security of the project depends on these contracts. While we are unaware of any immediate issues, it is important to note that DeFi protocols may be vulnerable to market manipulation, computational errors, etc. Furthermore, we also want to note that the project assumes constant supply of the external tokens, i.e., inflationary and deflationary tokens are not compatible (e.g., with `Fodl/modules/FundsManager/FundsManager.sol`).

Recommendation: We recommend reviewing external contracts to make sure they work as expected. Furthermore, document any assumptions regarding token compatibility.

QSP-7 Privileged Roles and Ownership

Severity: Informational

Status: Acknowledged

Description: Smart contracts will often have `owner` variables to designate the person with special privileges to make modifications to the smart contract. Specifically:

1. The contracts use the diamond pattern, which allows the owner to change implementations arbitrarily.
2. The `FundsManager` implements a tax on withdrawals. There is no limit in the contract for how large the tax percentage can be.

Recommendation: This centralization of power needs to be made clear to the users, especially depending on the level of privilege the contract allows to the owner.

QSP-8 Inaccurate `BLOCKS_PER_YEAR`

Severity: Informational

Status: Acknowledged

File(s) affected: `Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol`, `Fodl/modules/Lender/Aave/AaveLendingAdapter.sol`

Description: The constant `BLOCKS_PER_YEAR = 365 * 24 * 60 * 4` suggests 15 second block times, which does not reflect the typical block times for the past year of around 13 seconds. This will inflate the calculated supply and borrow APRs. Furthermore, the actual number of blocks may depend on the network on which the contracts are deployed.

Recommendation: Update the value in both files. Consider adding a setter function to allow updating the variable if average block times significantly change.

QSP-9 Constraints do not match inline comments

Severity: Informational

Status: Fixed

File(s) affected: `Fodl/connectors/SimplePosition/SimplePositionStopLossConnector.sol`, `Fodl/modules/StopLoss/StopLossStorage.sol`

Description:

1. In `StopLossStorage.sol` on L25, the `unwindFactor` is claimed to range from “1 to 100”. However, `SimplePositionStopLossConnector.configureStopLoss()` checks that `unwindFactor <= MANTISSA`. A similar issue exists for `slippageIncentive`.
2. In `SimplePositionStopLossConnector.sol` on L52,54:

```
// check that collateralUsageLimit <= collateralUsage

require(getCollateralUsageFactor() > stopLossConfiguration.collateralUsageLimit, 'SLC5');
```

Either the first line should use `<=`, or the second line should use `>=` to match the comment. Note that L161 uses the same constraint.

Recommendation: Ensure that each code snippet precisely matches the inline documentation.

QSP-10 Clone-and-Own

Severity: Informational

Status: Fixed

File(s) affected: `contracts/libs/UniswapV3Utils.sol`, `Libs/IWETH.sol`, `Libs/IERC20WithMetadata.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. Rather than the clone-and-own approach, a good industry practice is to use a framework for managing library dependencies. This eliminates the clone-and-own risks yet allows for following best practices, such as, using libraries.

Recommendation: `IUniswapV3Pool` could be replaced by import from `UniswapV3` github interface. Similar recommendations apply to `IWETH.sol` and `IERC20WithMetadata.sol`.

QSP-11 Unclear use of `create2` opcode

Severity: Undetermined

Status: Fixed

File(s) affected: `contracts/core/FoldingRegistry.sol`

Description: `create2` is used in the context where there is a need to predict the contract address before deploying it. There is no mention in the documentation for such a feature, yet, the opcode is still used in `createAccount()` function.

Recommendation: If there is no reason to use `create2`, then, perhaps, it is better to create new contracts the regular way.

Update: Added reasoning on why the team uses `create2` (also explained in QSP-2).

Automated Analyses

Slither

Slither reported the following:

1. Eth transfers to arbitrary user in:
    1. `Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol#134`,
    2. `Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol#1531`,
    3. `Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol#1681`,
    4. `Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol#181`,
    5. `Libs/StopLossExecutor.sol#48`.
    We classified these issues as false positives.

2. **Update:** fixed. State variable shadowing where `Fodl/connectors/SimplePosition/SimplePositionLeveragedLendingConnector.sol#34` shadows `FundsManager.MANTISSA` defined in `Fodl/modules/FundsManager/FundsManager.sol#16`.
3. Multiplication on the result of a division in:
    1. `FundsManager.withdraw(uint256,uint256)` due to `contracts/Fodl/modules/FundsManager/FundsManager.sol#41` being used in `contracts/Fodl/modules/FundsManager/FundsManager.sol#46`. **Update:** acknowledged.
    2. `CompoundForksLendingAdapter.getCollateralUsageFactor(address)` due to `contracts/Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol#72` being used in `contracts/Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol#74`. **Update:** fixed.
    3. `Mathemagic.mulDiv(uint256,uint256,uint256)` due to `contracts/Libs/Mathemagic.sol#31` being used in `contracts/Libs/Mathemagic.sol#35-43`. **Update:** acknowledged.
    We recommend looking into these issues.

4. Uses of strict equality in `Fodl/core/FoldingRegistry.sol#82-84`, which we classified as a false positive.
5. Contract locking ether since `contracts/Fodl/core/FoldingAccount.sol` has a payable function `FoldingAccount.fallback()` but does not have a function to withdraw the ether. We recommend adding a withdraw function in this case.
6. Uninitialized local variables in `Libs/StopLossExecutor.sol` lines 113-118. We consider the issue a false positive.
7. Ignored return values in:
    1. `Fodl/modules/Lender/Aave/AaveLendingAdapter.sol` lines 89, 122, 132,
    2. `Fodl/modules/Lender/LendingDispatcher.sol` lines 80, 99, 108, 117-119, 128-130,
    3. `Libs/StopLossExecutor.sol`, lines 82, 92, 99, 102, 112-128.
    We recommend checking the return values.

Code Documentation

1. The code could use better inline docs throughout. A number of functions has no comments. In most cases, the expected precision/decimals of variables is undocumented. As one example, `getReferencePrice()` units are undocumented and differ based on the platform.
2. In `Fodl/modules/FoldingAccount/FoldingAccountStorage.sol` on L2-14, the comment “This is the caller of the account. It is different from 0 only during a

transaction. It's value is set and unset by the `_fallback` function in FoldingAccount" should instead say "...unset by the `delegate` function...". **Update:** acknowledged.

3. The contract `Fodl/modules/FundsManager/FundsManager.sol` could use more documentation. For example, it is unclear that `principle` and `profit` are intended as percentages, and `holder` is a FODL-owned account for fee-collection purposes. The `subsidy` calculation could also use more documentation describing its rationale. **Update:** acknowledged.
4. In `connectors/SimplePosition/SimplePositionStopLossConnector.sol`, there are several docstrings that state "see ISimplePositionStopLossConnector.sol". However, there is no additional documentation in the interface file.

Adherence to Best Practices

1. **Update:** fixed. Unused `ABIEncoderV2` pragma, e.g., in the following files:
    1. `connectors/SimplePosition/SimplePositionBaseConnector.sol`,
    2. `connectors/SimplePosition/SimplePositionLendingConnector.sol`,
    3. `Fodl/modules/FundsManager/FundsManager.sol`,
    4. `Fodl/modules/FlashLoaner/DyDx/DydxFlashloanBase.sol`.
2. **Update:** fixed. Unresolved TODO items in:
    1. `connectors/SimplePosition/SimplePositionLeveragedLendingConnector.sol#155`,
    2. `Fodl/core/FoldingRegistry.sol#114`,
    3. `Fodl/modules/Lender/Aave/AaveLendingAdapter.sol#102`,
    4. `Fodl/modules/Lender/Aave/AaveLendingAdapter.sol#113`.
    5. `Fodl/modules/Exchanger/SushiswapExchangerAdapter.sol#12`,
3. **Update:** fixed. Old code still present in `connectors/SimplePosition/SimplePositionStopLossConnector.sol#56-57`.
4. **Update:** acknowledged. Several libraries/interfaces appear to be unused and could be removed: `ForceSend.sol`, `IDAI.sol`, `IDOLA.sol`.
5. **Update:** fixed. In `Fodl/modules/StopLoss/StopLossStorage.sol`, the `fodlFee` variable mentioned in the comments does not appear to exist. This relates to L55,56 of `connectors/SimplePosition/SimplePositionStopLossConnector.sol`, which have commented out fee computations related to `fodlFee`.
6. **Update:** acknowledged. `Fodl/modules/Exchanger/TestExposedExchangerDispatcher.sol` may be better located in the mocks directory.

Test Results

Test Suite Results

The test suite executed successfully.

```
Aave Lender Module
  Constructor
     ✓ sets addresses correctly
     ✓ reverts when intialising with 0 address
  getReferencePrice
     ✓ returns correct value for WETH
     ✓ returns correct value for WBTC
     ✓ returns correct value for USDC
  supply and borrow
     ✓ can supply DAI collateral
     ✓ should not be able to supply not supported token
     ✓ should be able to borrow dai
     ✓ should be able to repay borrowed dai
     ✓ can supply USDT collateral
     ✓ should be able to borrow usdt
     ✓ should be able to repay borrowed usdt
     ✓ should be able to redeem supplied dai
     ✓ should be able to redeem supplied usdt
  getCollateralFactorForAsset()
     ✓ correctly retrieves collateral factor for ETH

AAVE platform metadata
   ✓ Returns correct data for DAI reserves

ClaimRewardsConnector
  claimRewards()
     ✓ cannot claim rewards before initialising simple position
     ✓ can claim rewards after initialising simple position

CompoundForksLendingAdapter
  constructor()
     ✓ sets immutables correctly
     ✓ reverts when intialising with 0 address
  enterMarkets()
     ✓ can enter valid markets
     ✓ cannot enter when one invalid market
  supply() and borrow()
     ✓ when markets not entered, cannot borrow
     ✓ when markets entered, can supply and borrow
     ✓ cannot supply more WETH than balance
     ✓ can supply USDT
     ✓ cannot supply more USDT than balance
  redeem()
     ✓ can redeem small amount
     ✓ cannot redeem more than collateral
  repay()
     ✓ can repay less than borrow balance
     ✓ cannot repay more than borrowBalance
  claimRewards()
     ✓ can claim rewards
  getSupply/BorrowBalance()
     ✓ can get supply balance
     ✓ can get borrow balance
  getReferencePrice()
     ✓ returns correct value for USDC
     ✓ returns value in expected deviation range 2% for DAI
     ✓ returns value in expected deviation range 2% for USDT
  getCollateralUsageFactor()
     ✓ can get collateralUsageFactor when no supply and borrow
     ✓ can get collateralUsageFactor when only supply
     ✓ can get collateralUsageFactor when both supply and borrow
  getCollateralFactorForAsset()
     ✓ correctly retrieves collateral factor for ETH
  getAssetMetadata()
     ✓ Returns correct data for DAI reserves

ControlledExchanger
  [WBTC, USDC] with price updates: 1000000000000000000,1000000000000000000
     ✓ getAmountsIn
     ✓ getAmountsOut
     ✓ swapExactTokensForTokens
     ✓ swapTokensForExactTokens
  [WBTC, USDC] with price updates: 1400000000000000000,1000000000000000000
```

```
      ✓ getAmountsIn
      ✓ getAmountsOut
      ✓ swapExactTokensForTokens
      ✓ swapTokensForExactTokens

ExchangerDispatcher
   ✓ dispatches arguments to mock
   ✓ forwards return value from mock
   ✓ reverts when calling non contract
   ✓ reverts when calling non contract

Test FodlNFT
   ✓ should initialize properly
  mint nft account
     ✓ owner can mint
     ✓ non-owner cannot mint
     ✓ cannot mint duplicate
     ✓ should set owner properly

FodlToken
   ✓ Assigns initial balance
   ✓ Transfer adds amount to destination account
   ✓ Transfer emits event
   ✓ Cannot transfer above the amount
   ✓ Cannot transfer from empty account
   ✓ Can transferAndCall when receiver is contract
   ✓ Can transferAndCall when received is not contract

Test FoldingRegistry
  Initialisation and upgrades
     ✓ should initialize properly
     ✓ updates version
     ✓ cannot initialize twice
     ✓ updates functions
     ✓ allows new storage and behaves like V1
     ✓ cannot receive ether
  create account
     ✓ should create new
     ✓ should set owner properly
     ✓ allows to predict new account address per owner
     ✓ transferring an account to another owner does not affect the prediction of new accounts
  implementations
     ✓ should link signatures to implementation
     ✓ should overwrite implementation for existing signatures
     ✓ should remove implementation when it exists
     ✓ should revert when FR2
  platforms
     ✓ should add platform with adapter
     ✓ should revert when trying to add same platform twice
     ✓ should revert when trying to change platform adapter of non existent platform
     ✓ should change adapter for existing platform
     ✓ should revert when trying to remove non existing platform
     ✓ should remove existing platform
     ✓ should get platform adapter for given platform
     ✓ should revert when lending adapter not found
     ✓ should add platforms for an adapter in a batch
     ✓ should revert when no platforms passed to add in a batch
  CToken
     ✓ should add cToken to platform
     ✓ should get token from platform
     ✓ should revert when trying to add token that already exists on given platform
     ✓ should remove token from platform
     ✓ should revert when cToken mapping not found
     ✓ should revert when trying to remove non existent token from platform
  Exchanger
     ✓ should add exchange with adapter
     ✓ should revert when trying to add exchanger twice
     ✓ should get exchanger
     ✓ should revert when exchanger not found
     ✓ should change exchanger for an existing flag
     ✓ should remove exchanger
     ✓ should revert when trying to remove non existent exchanger
  Access Control
     ✓ only owner can modify configurations

FoldingRegistryUpgradability
   ✓ updates version
   ✓ updates functions
   ✓ allows new storage and behaves like V1

LendingPlatformLens
   ✓ Initializes with registry address
   ✓ Gets a token meta from AAVE
   ✓ Gets 2 tokens meta from AAVE
   ✓ Gets a token meta from Compound

  ResetAccountConnector
    resetAccount()
       ✓ is called on token transfer to new owner
       ✓ cannot be called by anyone except fodlNFT

  MerkleDistributor
    token()
       ✓ returns the token address
    merkleRoot
       ✓ returns the zero merkle root
       ✓ can set the merkle root
       ✓ cannot set merkle root without pausing
       ✓ cannot set same merkle root twice
    claim
       ✓ claiming is paused before root publishing
      two account tree
0x5d6fa53301c08693af8396874b6abd62f13e51b9cdb7753c013aa4a27bbe4f94
         ✓ claiming is allowed after setting root
0x5d6fa53301c08693af8396874b6abd62f13e51b9cdb7753c013aa4a27bbe4f94
         ✓ fails for empty proof
0x5d6fa53301c08693af8396874b6abd62f13e51b9cdb7753c013aa4a27bbe4f94
         ✓ successful claim
0x5d6fa53301c08693af8396874b6abd62f13e51b9cdb7753c013aa4a27bbe4f94
         ✓ transfers the token
0x5d6fa53301c08693af8396874b6abd62f13e51b9cdb7753c013aa4a27bbe4f94
         ✓ cannot claim same root twice
0x5d6fa53301c08693af8396874b6abd62f13e51b9cdb7753c013aa4a27bbe4f94
         ✓ cannot claim once paused
    parseBalanceMap
       ✓ all claims work

  SimplePositionLendingConnector
    increaseSimplePositionWithFunds()
       ✓ can only supply
       ✓ can only borrow
       ✓ can supply and borrow
    decreaseSimplePositionWithFunds()
       ✓ cannot be called when position not initialised
       ✓ can only redeem supply
       ✓ can only repay borrow
       ✓ can redeem supply and repay borrow
    claimRewards()
       ✓ cannot claim rewards before initialising simple position
       ✓ can claim rewards after initialising simple position

SimplePositionLeveragedLendingConnector
  increasePositionWithFlashLoan
     ✓ allows to open a leveraged position with random parameters
     ✓ reverts when supplyAmount > principal + flashLoanAmount
     ✓ allows to increase leverage after opening a position
     ✓ allows to use different tokens for flashloan
     ✓ cannot open when borrowAmount is lower than flashRepaymentAmount
     ✓ reverts if called by non owner
  decreasePositionWithFlashLoan
     ✓ reverts when called by non-owner
    when having a previous position
       ✓ allows to unwind position completely by repaying borrow, flashloan token = borrow token
       ✓ allows to unwind position completely by repaying borrow, flashloan token = supply token
       ✓ cannot send the principal to owner if not enough redeemed when flashloan token = supply token
       ✓ correctly taxes profit when flashloan token = supply token
       ✓ cannot repay borrow if repayBorrowAmount is higher than flashLoanAmount when flashToken = borrowToken
       ✓ cannot repay borrow if repayBorrowAmount is higher than flashLoanAmount when flashToken = supplyToken
       ✓ allows to partially decrease position while maintaining leverage when flashToken = borrowToken
    without having a previous position
       ✓ reverts

SimplePositionStopLossConnector
  configureStopLoss()
    when position has not been opened yet
       ✓ reverts
    when position is open but configured limit is over current collateral use
       ✓ reverts
    when position is open
       ✓ allows to change stop loss parameters
       ✓ reverts if unwindFactor is over 100% (1e18)
       ✓ reverts if slippageIncentive is over 100% (1e18)
       ✓ reverts if collateralUsageLimit is over 100% (1e18)
       ✓ reverts if called by non owner
  stop loss reset
     ✓ allows the fodlNFT contract to reset on NFT transfer
```

```
    executeStopLoss()
      reverts
         ✓ when position is not yet defined
         ✓ reverts if unwind factor is not configured
      works correctly
         ✓ reverts if borrow usage limit is under the configured threshold
         ✓ allows to trigger stop loss when collateral usage goes above limit
         ✓ allows to trigger stop loss when collateral usage goes above limit, full unwind

  UniswapExchangerAdapter
    UniswapExchangerAdapter
      constructor()
         ✓ sets immutables correctly
         ✓ reverts if passing address 0x0 as UNI_ROUTER address
      exchange()
         ✓ swaps the amount and resets allowance
         ✓ gets amount of output token for given amount of input token
         ✓ gets amount of input token required to get given amount of output token
    SushiswapExchangerAdapter
      constructor()
         ✓ sets immutables correctly
         ✓ reverts if passing address 0x0 as UNI_ROUTER address
      exchange()
         ✓ swaps the amount and resets allowance
         ✓ gets amount of output token for given amount of input token
         ✓ gets amount of input token required to get given amount of output token

  173 passing (7m)
```

Code Coverage

Overall, the code features very good coverage.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
Fodl/connectors/	100	100	100	100	
AllConnectors.sol	100	100	100	100	
ResetAccountConnector.sol	100	100	100	100	
Fodl/connectors/SimplePosition/	87.58	86.84	86.36	87.65	
ClaimRewardsConnector.sol	100	87.5	100	100	
SimplePositionBaseConnector.sol	47.83	50	71.43	47.83	… 68,69,82,83
SimplePositionLendingConnector.sol	100	91.67	100	100	
SimplePositionLeveragedLendingConnector.sol	95.45	84.21	100	95.45	75,120,195
SimplePositionStopLossConnector.sol	85.29	93.75	75	85.71	… 115,116,117
Fodl/connectors/interfaces/	100	100	100	100	
IClaimRewardsConnector.sol	100	100	100	100	
IResetAccountConnector.sol	100	100	100	100	
ISimplePositionBaseConnector.sol	100	100	100	100	
ISimplePositionLendingConnector.sol	100	100	100	100	
ISimplePositionLeveragedLendingConnector.sol	100	100	100	100	
ISimplePositionStopLossConnector.sol	100	100	100	100	
Fodl/core/	100	81.48	100	100	
FodlNFT.sol	100	100	100	100	
FoldingAccount.sol	100	87.5	100	100	
FoldingRegistry.sol	100	79.55	100	100	
Fodl/core/interfaces/	100	100	100	100	
ICTokenProvider.sol	100	100	100	100	
IExchangerAdapterProvider.sol	100	100	100	100	
IFoldingAccountOwnerProvider.sol	100	100	100	100	
IFoldingConnectorProvider.sol	100	100	100	100	
ILendingPlatformAdapterProvider.sol	100	100	100	100	
Fodl/lens/	53.33	50	60	53.33	
LendingPlatformLens.sol	100	50	100	100	
SimplePositionLens.sol	0	100	0	0	… 27,28,30,31
Fodl/mocks/	48.28	100	56.52	48.28	
AavePriceOracleMock.sol	40	100	57.14	40	… 36,38,42,46
CompoundPriceOracleMock.sol	**100**	**100**	**100**	**100**	
ERC20Mock.sol	0	100	0	0	10,14,15

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
ExchangerMock.sol	33.33	100	33.33	33.33	... 55,56,65,66
FoldingRegistryUpgradedMock.sol	100	100	100	100	
Fodl/modules/Exchanger/	95.83	62.5	95.45	95.83	
ControlledExchanger.sol	100	50	100	100	
ControlledExchangerAdapter.sol	100	100	100	100	
ExchangerDispatcher.sol	80	100	80	80	58,61
IExchanger.sol	100	100	100	100	
IUniswap.sol	100	100	100	100	
SushiswapExchangerAdapter.sol	100	100	100	100	
TestExposedExchangerDispatcher.sol	100	100	100	100	
UniswapExchangerAdapter.sol	100	100	100	100	
Fodl/modules/FlashLoaner/DyDx/	97.06	60	100	97.22	
DyDxFlashModule.sol	100	50	100	100	
DydxFlashloanBase.sol	92.31	100	100	92.31	31
ICallee.sol	100	100	100	100	
ISoloMargin.sol	100	100	100	100	
Fodl/modules/FoldingAccount/	100	100	100	100	
FoldingAccountStorage.sol	100	100	100	100	
Fodl/modules/FundsManager/	95.24	60	100	95.24	
FundsManager.sol	95.24	60	100	95.24	58
Fodl/modules/Lender/	100	100	100	100	
ILendingPlatform.sol	100	100	100	100	
LendingDispatcher.sol	100	100	100	100	
Fodl/modules/Lender/Aave/	100	100	100	100	
AaveLendingAdapter.sol	100	100	100	100	
Interfaces.sol	100	100	100	100	
Fodl/modules/Lender/Compound/	100	91.67	100	100	
CompoundForksLendingAdapter.sol	100	91.67	100	100	
ICEther.sol	100	100	100	100	
ICToken.sol	100	100	100	100	
ICompoundPriceOracle.sol	100	100	100	100	
IComptroller.sol	100	100	100	100	
Fodl/modules/SimplePosition/	100	66.67	100	100	
SimplePositionStorage.sol	100	66.67	100	100	
Fodl/modules/StopLoss/	100	100	100	100	
StopLossStorage.sol	100	100	100	100	
FodlRewards/	100	100	100	100	
RewardsDistributor.sol	100	100	100	100	
FodlToken/	100	100	100	100	
FodlToken.sol	100	100	100	100	
Libs/	94.74	62.5	100	100	
ForceSend.sol	100	100	100	100	
IDAI.sol	100	100	100	100	
IDOLA.sol	100	100	100	100	
IUSDC.sol	**100**	**100**	**100**	**100**	
IUSDT.sol	100	100	100	100	

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
IWBTC.sol	100	100	100	100	
IWETH.sol	100	100	100	100	
Mathemagic.sol	92	50	100	100	
Uint2Str.sol	100	100	100	100	
All files	**92.36**	**82.3**	**90.96**	**92.86**	

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

74dc13ff718e3f83c7977a28e79865dece699961089918f2ea8286a5b5a0b7a4 ./contracts/FodlRewards/RewardsDistributor.sol

ef6f78578a9a7d6bef09e9172acad20c50bf17a601e776a41d0049e9321b33ae ./contracts/FodlToken/FodlToken.sol

947ad315b2475e0a5d2ab5be181baf983e047f95b3bb270cbf15d90dc0b77e9a ./contracts/Libs/ForceSend.sol

7b1a5f1774514bc2d6b92001e2c2c34422d440d4145298b71303b50227c7c895 ./contracts/Libs/IWETH.sol

4784d10deb20c4bd007180b62716ecd3f4e7b9959e1a74916d58f2eaf12dcb9c ./contracts/Libs/IDOLA.sol

9dc0781705e664521d39385bf7fec7896f790a6066d278e954cd9ec358f8efde ./contracts/Libs/Mathemagic.sol

fab7bb7fcafe611b18dbeb31d63f48f25bfac2162594a0cca7a8c3014a9968a8 ./contracts/Libs/IUSDC.sol

6e7d3d89e6681915ccf9511008e08bc429d9942dd8c1886a044d330264d506c6 ./contracts/Libs/IUSDT.sol

d20becaeb1d803681a24a8deeaa002c29963b95f4ac4441d8c5144eb59a39d4f ./contracts/Libs/IDAI.sol

67061c49aa02ce7960e2ea33053730d9028698972e3ff82dd231e3d47270ff2b ./contracts/Libs/IWBTC.sol

6c81a6ebcf1358f009b73aa38ba780d39a2ae246fbdc270bec46b166973cea90 ./contracts/Libs/Uint2Str.sol

d665de3bc2add5cb273e4569a7021b07d4d2b4a55b5868bdfdc69e1786ab0182 ./contracts/Fodl/modules/FundsManager/FundsManager.sol

64cdaf00247e6ad477e7d6a93727c6fe9871314b1885676669f94df304618856 ./contracts/Fodl/modules/FlashLoaner/DyDx/DydxFlashloanBase.sol

857de365c32a455bd57729c417d65fd1a7c75a2ea6908c179da804b27902bc8b ./contracts/Fodl/modules/FlashLoaner/DyDx/DyDxFlashModule.sol

c4cd5e0e7fcab88a8566b47ae8cb7c1ee90cdbc541494fe86ae2e63828b29fd2 ./contracts/Fodl/modules/FlashLoaner/DyDx/ICallee.sol

f540c9f99e2b5fdf0a49de29c9c29a3b9a75418ba626fd9eeecdeeb39dfc62d8 ./contracts/Fodl/modules/FlashLoaner/DyDx/ISoloMargin.sol

70fb670431ebbf30b319816cb67c18acbeac058afdc4e995827d3e9fba30d7b9 ./contracts/Fodl/modules/StopLoss/StopLossStorage.sol

834d40828daf0484000e387316abdb70c7863bf403842d5b6d6cb40d0f972fcd ./contracts/Fodl/modules/SimplePosition/SimplePositionStorage.sol

e5685d3e01eb5d4518f9e54bfe57e869f52a73bedc686afebfbf9b6d2ec17cdb ./contracts/Fodl/modules/Exchanger/ControlledExchangerAdapter.sol

b1c371c56fa2a4e753513fbd72d41c722b436b39e72397771c152144d741dd60 ./contracts/Fodl/modules/Exchanger/IExchanger.sol

c8a276f762da0c39604d39c88d2958e0b33e4b600a48f092229643f3014729ad ./contracts/Fodl/modules/Exchanger/SushiswapExchangerAdapter.sol

755e87d708ecb6905998a20a4ba3ef4b3b4ab314f188e33d8c02f837fca64992 ./contracts/Fodl/modules/Exchanger/TestExposedExchangerDispatcher.sol

ced9f858d556b35e6dc95b354e3c7d06759066a787be17f4a13d2d7e07b29891 ./contracts/Fodl/modules/Exchanger/UniswapExchangerAdapter.sol

3782c4ef6da6bd343a289434b8f7b2b0fc187f94649bd1890cabbf2e57880b39 ./contracts/Fodl/modules/Exchanger/ControlledExchanger.sol

9da91698040745856412aafc2715d9d0b7fceb2e648e5385566ac62c672b4052 ./contracts/Fodl/modules/Exchanger/IUniswap.sol

df993f79650164172fe8c38011dd9ffefa52ed1c0e36b08849f8a1e87bb88f25 ./contracts/Fodl/modules/Exchanger/ExchangerDispatcher.sol

967cb651bf09fc1f6998a2b406e4bf930a97fedbcbfc663b8eaf1fbdf6c759dc ./contracts/Fodl/modules/Lender/ILendingPlatform.sol

f677eafe185e323dce00c49fdaabbce5e07046d5dac8d1f0122c8f45fcb84da4 ./contracts/Fodl/modules/Lender/LendingDispatcher.sol

6a67ec7b910188ba8a8ddd150a97faccc3279fd46cdf563c656d9f5536d72c0d ./contracts/Fodl/modules/Lender/Compound/ICToken.sol

090dcfd357daf5161ac537646d5aff820133d778ea84201987e4944c23024b77 ./contracts/Fodl/modules/Lender/Compound/ICompoundPriceOracle.sol

b9e00118d975f794b4947621f85adb58c7f7f9e02abf51abd50c42d0f711a867 ./contracts/Fodl/modules/Lender/Compound/IComptroller.sol

132d1f83b08e4e90080bfbf6d2b3c3ba97f79ca6a7f38479f02376604eaae9fb ./contracts/Fodl/modules/Lender/Compound/CompoundForksLendingAdapter.sol

79048963eba663389230fa2d8dd22d6f1342f7986831878e5c0dffcb11fd2019 ./contracts/Fodl/modules/Lender/Compound/ICEther.sol

6a1f647c14370e71a997cdbc72aac45155dbddcce6b0513fbaf6140b3e36dd83 ./contracts/Fodl/modules/Lender/Aave/AaveLendingAdapter.sol

df78f554c73737bee8d29e56b760f4e50192b0a144f6311bf6fadedbb307689b ./contracts/Fodl/modules/Lender/Aave/Interfaces.sol

5e7ba7019ed10d057b8bf5632e73717eb7646752968b454a7c18d91e3b5245b2 ./contracts/Fodl/modules/FoldingAccount/FoldingAccountStorage.sol

a5f4569f74e6b70804b50284e89621ee45eae189e5639a62f7eb8da9ae8d3321 ./contracts/Fodl/lens/SimplePositionLens.sol

7318f2b43930ea247799000724fcdf9fa470fcd9bc04800e5b1b2f597ddaca0d ./contracts/Fodl/lens/LendingPlatformLens.sol

cd4e64e1bdf6161f178d92d3cd4f2f416f459cbabba3939184cf8c515c3e6f56 ./contracts/Fodl/core/FoldingAccount.sol

696f0b46bb6c60eefd82da4bc7049621ff075f94ae93c9809d46f4fdfc372e01 ./contracts/Fodl/core/FoldingRegistry.sol

dfc53f98d26b1e5ba4796d1d6de7a0b47fdede5b62e75f62f997879d6a4e3255 ./contracts/Fodl/core/FodlNFT.sol

97cf28bd354b751a322164907fcf12671cebfaac047431001efdfa98462f58a7 ./contracts/Fodl/core/interfaces/IFoldingAccountOwnerProvider.sol

b6e2b4aec88bd63093472b515f1738fccb94e2b5180f16e94b4430f131ccf7a1 ./contracts/Fodl/core/interfaces/IFoldingConnectorProvider.sol

330ee2437a9718d480bbee2a279a6e746803cbe99a05ecd1fa5a87366a3fdb01 ./contracts/Fodl/core/interfaces/ICTokenProvider.sol

599407b00fed7c25723b84d27bd1cfbe1c8c7c24c81f300df59462d99fee6c4b ./contracts/Fodl/core/interfaces/IExchangerAdapterProvider.sol

fde977c249396e5ccd1d880769971238c81b96ad16f5938bd0e35a14b9cce0f8 ./contracts/Fodl/core/interfaces/ILendingPlatformAdapterProvider.sol

7a5c964d7c9ac958c2cd4f748654a58cc0b44487be16905c16f0b577de7583ec ./contracts/Fodl/connectors/AllConnectors.sol

7fb4e88e8b126b1a9ffa7d5b0adbebb77225d70c30ccb657431100c360f0032a ./contracts/Fodl/connectors/ResetAccountConnector.sol

2a8d6f253b83282c45e1030da11ed996af2d6e79348b795dadfd682e1a6290c0 ./contracts/Fodl/connectors/interfaces/IResetAccountConnector.sol

6e6b1e0ba9ae305dcf776809b299fc8df8707f1ac5286edb9d95b3533734a20c ./contracts/Fodl/connectors/interfaces/ISimplePositionBaseConnector.sol

03040c7553775115897c4312313f30ad113a173c2a31e26cd4b10bfeaadec574 ./contracts/Fodl/connectors/interfaces/ISimplePositionLendingConnector.sol

67c145fb9d431fde8e82493018ea84e85b275852b205ae8d218f6b2fdf8691d2
./contracts/Fodl/connectors/interfaces/ISimplePositionLeveragedLendingConnector.sol

d9ca2126cd628ad2ea058e8426aa7ac54c846b83509347f1bfc726a0fdb79e06 ./contracts/Fodl/connectors/interfaces/IClaimRewardsConnector.sol

39f65c0cd546bd2dee5ab42ab570d1ba55aab29275977630aca7e298d8fbdf96 ./contracts/Fodl/connectors/interfaces/ISimplePositionStopLossConnector.sol

50f557c8675ea8f2c5e0b48252ae729cf7567aa54604215cc9cbb432ff97f380 ./contracts/Fodl/connectors/SimplePosition/ClaimRewardsConnector.sol

b81a9696b1ed7187c384ce37a75d1e15296b7d4787637c3a4422198568f66f74 ./contracts/Fodl/connectors/SimplePosition/SimplePositionBaseConnector.sol

c69c1aa96345ae04203bba8a5557dfceade73e1c2a93b1199794ad7a6a7b7fb4
./contracts/Fodl/connectors/SimplePosition/SimplePositionLeveragedLendingConnector.sol

7bf37e8a73720ae08fd37050e39faccddfdd0e2b0c52cf9371416443786e5a69 ./contracts/Fodl/connectors/SimplePosition/SimplePositionLendingConnector.sol

4884c42fd1b7c54fa7e6bd79d1595463d7012343909ee0f7bdec45a3087df133
./contracts/Fodl/connectors/SimplePosition/SimplePositionStopLossConnector.sol

dc20d4dddc3de1d6ecc4a5b026584c397e9d722a4eeb2e04610a6ad980447c70 ./contracts/Fodl/mocks/CompoundPriceOracleMock.sol

f0d31d253eb0a407f70a7dac768f13bc1afb9c5a8855b5f110bef43fae2cb70b ./contracts/Fodl/mocks/FoldingRegistryUpgradedMock.sol

8cf4fb7700a7b2ee840aa2d1d35e9dedbb8e8273a7cf667c972e7717e3ba5db9 ./contracts/Fodl/mocks/ERC20Mock.sol

d4e7733d76a7e628529e29d5397bf844fe92207f632b55c1e63b2a39630fc6e4 ./contracts/Fodl/mocks/ExchangerMock.sol

c38104079f0e8bc8861614cfbc20e5f4570dad2df62035af67138e96dc950edb ./contracts/Fodl/mocks/AavePriceOracleMock.sol

Tests

e3100acb813c8be66771bba88cf92f029fb25b0859fe1259c92755f5fa6adc9b ./test/ClaimRewardsConnector.test.ts

38f93e283dea9a02f640e8969aad5523e0bd72fa203d6249fe5a4609c0052421 ./test/UniswapExchangerAdapter.test.ts

eeb006b49f5e58f61dfb40e137e4d9837485bf7f01a83285cd5008212d173b52 ./test/CompoundForksLendingAdapter.test.ts

4750fcdf436a5b3766fcb02645545cb425d7ea07252c973d0c136a81d35cbf5e ./test/AaveLendingAdapter.test.ts

604274ca288bf01223b55c48736078467e14335c9330be8947bfa294a3d4ec51 ./test/SimplePositionLeveragedLendingConnector.test.ts

8fb94e68e088b3f07f655de0b5325f121f8cab7a64cf4734a55e5668bc62ed8d ./test/RewardsDistributor.test.ts

b2f82aea085d1d5dc55f538ed1f1d9177ecda93fee19ddddbfaa04709a66be9d ./test/FoldingRegistryUpgradability.test.ts

f1054e48a8e8164025d448c48dcb527e71709550edf53e66e58f747bc9b06394 ./test/ControlledExchanger.test.ts

4a027000c72662cfcc3b66a1b84b5292acf6e87472917f4a995278f3bbe0bf31 ./test/LendingPlatformLens.test.ts

1a7cb1affdb0ddda5f12cb6d013ca6f8ac27b573cff87c8da0b72acb5f02216 ./test/AaveMetadata.test.ts

dc53b2b62d5429528bbf54a1ff28dbf19d654a5fb726b1d9b729dab27819c268 ./test/FoldingRegistry.test.ts

9c7f31adeb0ccaea2144da7805fcf3f65342265ae7673ac0454b3036e46cc037 ./test/SimplePositionLendingConnector.test.ts

25b43f69ee0c1939516cf02ae5e55731e66e02eeb600d696f3fa472da6ad030a ./test/SimplePositionStopLossConnector.test.ts

db1b3e10a87f15cb8df0d1de628b81eea642f53677e498a852b395387610a5be ./test/ResetAccountConnector.test.ts

cb3ce06d45ba1c606a44bcbdf23afd7c2a1ea0e1a5711f68d9654fc27122e6a7 ./test/FodlNFT.test.ts

f5e1f0946c1f2184fe87dc7f1c2000572a149936d138c79c310753758722de91 ./test/FodlToken.test.ts

e1a9b2906fb6aea0f94d035b752b22096224a030ef5a369b4837b705015b7d98 ./test/ExchangerDispatcher.test.ts

ba243f68e2266b35d72100d502c091f2a5442af42db229f2774fbc569fd2babe ./test/shared/fixtures.ts

6c97f56cd22a8a1349470990ae569a9d856681e6d44f1fdd60b0f68038fe4f52 ./test/shared/artifacts.ts

ff26e225138869354640158c8f546b7ad46856d68d42980f4b5e5b107f112d4a ./test/shared/errors.ts

6190256f55fcc9d3d45eb03a566133c0552fd18c568058e638a473f8eee36917 ./test/shared/utils.ts

f686331f52640ac880381c781f1434cae343b306f8e0b76ba0137fec19fbf9ab ./test/shared/constants.ts

Changelog

- 2021-09-17 - Initial report
- 2021-10-13 - Revised report based on commit `c56a53e`.

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.