



May 26th 2021 — Quantstamp Verified

Ribbon Finance

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

Executive Summary

Type	DeFi Options Strategies				
Auditors	Ed Zulkoski, Senior Security Engineer Kacper Bqk, Senior Research Engineer Jose Ignacio Orlicki, Senior Engineer				
Timeline	2021-05-17 through 2021-05-26				
EVM	Muir Glacier				
Languages	Solidity				
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review				
Specification	Theta Vault Design Doc				
Documentation Quality	<div><div></div>High</div>				
Test Quality	<div><div></div>Medium</div>				
Source Code	<table><tr><th>Repository</th><th>Commit</th></tr><tr><td>audit</td><td>0e08cc0</td></tr></table>	Repository	Commit	audit	0e08cc0
Repository	Commit				
audit	0e08cc0				

⚠ 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.
ℳ 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.

Total Issues	16 (0 Resolved)
High Risk Issues	0 (0 Resolved)
Medium Risk Issues	2 (0 Resolved)
Low Risk Issues	5 (0 Resolved)
Informational Risk Issues	5 (0 Resolved)
Undetermined Risk Issues	4 (0 Resolved)



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

Summary of Findings

During the audit 16 issues ranging in severity from Medium to Undetermined were noted. We recommend addressing all issues before deploying the code in production.

Note: The test suite was run from the main [ribbon-finance/structure-products](#) repository on commit [3fb63f1](#). The audit was performed on a different commit of the separate [ribbon-finance/audit](#) repository, and therefore discrepancies could exist between audited code and the code that will likely be used in production. We recommend carefully evaluating any differences between these two repositories when considering the contents of this report.

ID	Description	Severity	Status
QSP-1	<code>_setNextOption</code> has short 1 hour delay	^ Medium	Unresolved
QSP-2	Possibly stale price	^ Medium	Unresolved
QSP-3	Unchecked function arguments	✓ Low	Unresolved
QSP-4	<code>setAdapter</code> does not check if the adapter already exists	✓ Low	Unresolved
QSP-5	Withdrawal transaction ordering dependence (TOD) when there is little liquidity	✓ Low	Unresolved
QSP-6	Tolerated slippage may be too big	✓ Low	Unresolved
QSP-7	Unexpected Ether in <code>GammaAdapter</code>	✓ Low	Unresolved
QSP-8	Unlocked Pragma	○ Informational	Unresolved
QSP-9	Privileged Roles and Ownership	○ Informational	Acknowledged
QSP-10	Use of experimental <code>ABIEncoderV2</code>	○ Informational	Unresolved
QSP-11	External interactions	○ Informational	Unresolved
QSP-12	Unprotected <code>initialize</code> function	○ Informational	Unresolved
QSP-13	<code>setCap</code> may set the cap lower than the total balance	? Undetermined	Unresolved
QSP-14	Unused field in <code>OptionTerms</code>	? Undetermined	Unresolved
QSP-15	Possibly unnecessary stubbed functions	? Undetermined	Unresolved
QSP-16	No access control on <code>closeShort</code> may lead to DoS	? Undetermined	Unresolved

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](#) v0.7.1
- [Mythril](#) v0.22.19

Steps taken to run the tools:

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

Findings

QSP-1 `_setNextOption` has short 1 hour delay

Severity: *Medium Risk*

Status: Unresolved

File(s) affected: `RibbonThetaVault.sol`

Description: If the manager is malicious, users would have 1h (in the worst case scenario) before options expire. 1h may be insufficient if gas prices are high or the network is jammed. Further, this does not match the specification, which states: “In the event of a hack, depositors are able to have a 1 day notice to withdraw their tokens from the vault entirely.”

Recommendation: Set the delay to at least 1 day.

QSP-2 Possibly stale price

Severity: *Medium Risk*

Status: Unresolved

File(s) affected: `GammaAdapter.sol`

Description: In the functions `purchaseWithZeroEx` and `swapExercisedProfitsToUnderlying`, price feed oracles are used. However, when `latestRoundData` is invoked, the return values `updatedAt` and `answeredInRound` are not checked, and therefore the price data may be arbitrarily old.

Recommendation: Check that `updatedAt` and `answeredInRound` are recent.

QSP-3 Unchecked function arguments

Severity: *Low Risk*

Status: Unresolved

File(s) affected: `RibbonFactory.sol`

Description: The functions `initialize`, `setAdapter`, and `newInstrument` should check that each argument is non-zero.

Recommendation: Add the relevant `require` statements to each function.

QSP-4 `setAdapter` does not check if the adapter already exists

Severity: *Low Risk*

Status: Unresolved

File(s) affected: `RibbonFactory.sol`

Description: The function simply assigns the adapter without checking if it already exists. A malicious or uncareful owner may break existing functionality/adapters or pollute the `adapters` array.

Recommendation: Check if the adapter is already set for the specified protocol.

QSP-5 Withdrawal transaction ordering dependence (TOD) when there is little liquidity

Severity: *Low Risk*

Status: Unresolved

File(s) affected: `RibbonThetaVault.sol`

Description: The function `withdrawAmountWithShares` is used in withdrawals. It may fail if there isn't enough liquidity.

Recommendation: Inform users about this potential issue.

QSP-6 Tolerated slippage may be too big

Severity: *Low Risk*

Status: Unresolved

File(s) affected: `GammaAdapter.sol`

Description: When using Uniswap router in `GammaAdapter`, the default maximum slippage is 25% (receive at least `SLIPPAGE_TOLERANCE = 0.75` of what you want). This value seems too big and cannot be updated as it is a `constant`. This slippage is used when swapping USDC collateral for WETH in `swapExercisedProfitsToUnderlying()` and can help a flash-loan attack to be feasible when an attacker combines a call to `exercise()` with a manipulated WETH-USDC Uniswap pool.

Recommendation: Add a setter, that only the manager can use to update `SLIPPAGE_TOLERANCE` to an smaller value.

QSP-7 Unexpected Ether in `GammaAdapter`

Severity: *Low Risk*

Status: Unresolved

File(s) affected: `GammaAdapter.sol`

Description: Contract `GammaAdapter` can receive Ether as it has a function `receive() external payable {}`. But it does not seems designed to use sent Ether with the exception of `purchaseWithZeroEx()` that is receiving `msg.value > zeroExOrder.protocolFee` in 223-226. Also there is a check that the total satisfies `this(address).balance > zeroExOrder.protocolFee` in L270-273 which is redundant. This is redundant because `this(balance).balance >= msg.value`. You might also want to revert execution in `GammaAdapter.purchase()` as this function is `payable` but is just a stub. For `exercise()`, if this function has to be `payable` due to the interface, there is no check that `msg.value == 0`, and the same for other `payable` functions that are not using `msg.value`. This ether can get locked inside the contract if sent by mistake.

Recommendation: Clarify which of the following two alternatives you wan to adopt. 1. You want to use in `purchaseWithZeroEx()` existing Ether balance sent to the contract with `receive()`, in that case remove `require()` from L223-226 2. Alternatively, you only want in `purchaseWithZeroEx()` to use `msg.value` in `purchaseWithZeroEx()`, in that case you can remove `require()` from L270-273 and also revert in `receive()`.

QSP-8 Unlocked Pragma

Severity: *Informational*

Status: Unresolved

File(s) affected: `RibbonThetaVault.sol`, `GammaAdapter.sol`

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

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

QSP-9 Privileged Roles and Ownership

Severity: *Informational*

Status: Acknowledged

File(s) affected: `RibbonThetaVault.sol`, `GammaAdapter.sol`

Description: Smart contracts will often have `owner` variables to designate the person with special privileges to make modifications to the smart contract. In particular, the `manager` is a highly trusted role, which in the worst case could sell the tokens for as little as 1 wei.

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. We have marked this as acknowledged based on the contents of the [Theta Vault Design Doc](#).

QSP-10 Use of experimental `ABIEncoderV2`

Severity: *Informational*

Status: Unresolved

File(s) affected: `RibbonThetaVault.sol`, `GammaAdapter.sol`

Description: The pragma `ABIEncoderV2` is still considered experimental as of solidity version `0.7.2`.

Recommendation: If it is not in use we recommend removing it from the contracts.

QSP-11 External interactions

Severity: *Informational*

Status: Unresolved

File(s) affected: `GammaAdapter.sol`, `RibbonThetaVault.sol`

Description: The contracts rely upon several external protocols including Uniswap, Airswap, and Opyn.

Recommendation: We recommend vetting the external contracts to ensure that they work according to the assumptions. Please be aware of possible price manipulation, flash loan attacks, sandwich attacks that may result in market price manipulation.

QSP-12 Unprotected `initialize` function

Severity: *Informational*

Status: Unresolved

File(s) affected: `RibbonThetaVault.sol`

Description: The function `initialize` is unrestricted and could therefore be invoked by anyone after the contract is created, which may require redeployment.

Recommendation: Add access control to `initialize`, such as restricting it to the deployer's address.

QSP-13 `setCap` may set the cap lower than the total balance

Severity: *Undetermined*

Status: Unresolved

File(s) affected: `RibbonThetaVault.sol`

Description: It is not clear if this is intentional.

Recommendation: If this scenario is not desirable, add a `require` statement to `setCap` restricting it.

QSP-14 Unused field in `OptionTerms`

Severity: *Undetermined*

Status: Unresolved

File(s) affected: `IProtocolAdapter.sol`

Description: The field `paymentToken` is documented as "the token used to purchase the option." However, this field is not used anywhere in the project.

Recommendation: Clarify if the field should be used. Remove if not needed.

QSP-15 Possibly unnecessary stubbed functions

Severity: *Undetermined*

Status: Unresolved

File(s) affected: `GammaAdapter.sol`

Description:

1. The function `purchase` is "stubbed out for conforming to the `IProtocolAdapter` interface," however it will still allow users to invoke it (and lose sent ETH).
2. It is unclear why `receive` is needed in this contract.
3. It is unclear whether `premium` is correct as it always returns 0.

Recommendation: If the functions are not needed, remove them to reduce the attack surface. If the function is included purely for conforming to an interface, consider reverting to avoid any erroneous calls that may cause loss of funds.

QSP-16 No access control on `closeShort` may lead to DoS

Severity: *Undetermined*

Status: Unresolved

File(s) affected: `GammaAdapter.sol`

Description: The function `closeShort` may be invoked by any user. If the `GammaAdapter` or any contract that uses it via `delegateCall` does not properly restrict calls to `closeShort`, a DoS attack may be possible.
It is also unclear why `GammaAdapter` is not implemented as a library as opposed to a contract.

Recommendation: Add access control to the function as necessary. Consider defining `GammaProtocol` as a library.

Automated Analyses

Slither

Slither noted the following:

1. `RibbonFactory.burnGasTokens` does not check the return value of `chiToken.freeUpTo`.
2. `RibbonFactory.initialize` is unprotected and could potentially be called by anyone after deployment.
3. `GammaAdapter.purchaseWithZeroEx` ignores the return value of `router.swapETHForExactTokens`.

Mythril

Myth did not report any issues.

Adherence to Specification

The code generally adheres to the provided specification, except for the 1 hour delay issue as noted in QSP-1.

Code Documentation

The code is well documented.

Adherence to Best Practices

- The function `GammaAdapter.assetDecimals` is hard-coded for `USDC`. While this may be sufficient currently, `asset.decimals()` could be used to avoid future issues.

Test Results

Test Suite Results

```
#protocolName
  ✓ matches the protocol name
#nonFungible
  ✓ matches the nonFungible bool
#lookup0token
  ✓ looks up call oToken correctly
  ✓ looks up put oToken correctly
#swapExercisedProfitsToUnderlying
  ✓ swaps the exercised profit (273ms)
Call ITM
#premium
  ✓ has a premium of 0
#exerciseProfit
  ✓ gets exercise profit (49ms)
#purchaseWithZeroEx
  ✓ purchases with 0x exchange (144066ms)
  ✓ purchases twice (502ms)
  ✓ reverts when sellTokenAddress is not USDC
#exercise
  ✓ exercises otokens (15599ms)
#canExercise
  - can exercise
  ✓ cannot exercise before expiry
#createShort
  ✓ reverts when no matched oToken
  ✓ creates a short position (320ms)
#closeShort
  ✓ burns otokens and withdraws original amount before expiry (91ms)
  ✓ settles the vault and withdraws collateral after expiry (189ms)
#getOptionsAddress
  ✓ returns the correct otoken address
Call OTM
#premium
  ✓ has a premium of 0
#exerciseProfit
  ✓ gets exercise profit
#purchaseWithZeroEx
  ✓ purchases with 0x exchange (20814ms)
  ✓ purchases twice (250ms)
  ✓ reverts when sellTokenAddress is not USDC
#exercise
  ✓ exercises otokens
#canExercise
  - can exercise
  ✓ cannot exercise before expiry
#createShort
  ✓ reverts when no matched oToken
  ✓ creates a short position (147ms)
#closeShort
  ✓ burns otokens and withdraws original amount before expiry (91ms)
  ✓ settles the vault and withdraws collateral after expiry (141ms)
#getOptionsAddress
  ✓ returns the correct otoken address
Put OTM
#premium
  ✓ has a premium of 0
#exerciseProfit
  ✓ gets exercise profit
#purchaseWithZeroEx
  ✓ purchases with 0x exchange (20900ms)
  ✓ purchases twice (248ms)
  ✓ reverts when sellTokenAddress is not USDC
#exercise
  ✓ exercises otokens
#canExercise
  - can exercise
  ✓ cannot exercise before expiry
#createShort
  ✓ reverts when no matched oToken
  ✓ creates a short position (112ms)
#closeShort
  ✓ burns otokens and withdraws original amount before expiry (79ms)
  ✓ settles the vault and withdraws collateral after expiry (167ms)
#getOptionsAddress
  ✓ returns the correct otoken address

HegicAdapter
#protocolName
  ✓ matches the protocol name
#nonFungible
  ✓ matches the nonFungible bool
ETH CALL ITM
#premium
  ✓ gets premium of option
#purchase
  ✓ reverts when not enough value is passed
  ✓ reverts when buying after expiry
  ✓ reverts when passing unknown underlying
  ✓ creates options on hegic (74ms)
#exerciseProfit
  ✓ reverts when unknown options address passed
  ✓ gets correct exercise profit for an option (67ms)
#exercise
  ✓ exercises options with profit
  ✓ redirects exercise profit to recipient (44ms)
  ✓ reverts when past expiry
#canExercise
  ✓ can exercise
  ✓ cannot exercise twice
  ✓ cannot exercise after epxiry
ETH CALL OTM
#premium
  ✓ gets premium of option
#purchase
  ✓ reverts when not enough value is passed
  ✓ reverts when buying after expiry
  ✓ reverts when passing unknown underlying
  ✓ creates options on hegic (47ms)
#exerciseProfit
  ✓ reverts when unknown options address passed
```

```

    ✓ gets correct exercise profit for an option (63ms)
#exercise
    ✓ reverts when not ITM
    ✓ reverts when past expiry
#canExercise
    ✓ can exercise
    ✓ cannot exercise twice
    ✓ cannot exercise after expiry
ETH PUT ITM
#premium
    ✓ gets premium of option
#purchase
    ✓ reverts when not enough value is passed
    ✓ reverts when buying after expiry
    ✓ reverts when passing unknown underlying
    ✓ creates options on hegic (45ms)
#exerciseProfit
    ✓ reverts when unknown options address passed
    ✓ gets correct exercise profit for an option (55ms)
#exercise
    ✓ exercises options with profit (221ms)
    ✓ redirects exercise profit to recipient
    ✓ reverts when past expiry
#canExercise
    ✓ can exercise
    ✓ cannot exercise twice
    ✓ cannot exercise after expiry
ETH PUT OTM
#premium
    ✓ gets premium of option
#purchase
    ✓ reverts when not enough value is passed
    ✓ reverts when buying after expiry
    ✓ reverts when passing unknown underlying
    ✓ creates options on hegic (170ms)
#exerciseProfit
    ✓ reverts when unknown options address passed
    ✓ gets correct exercise profit for an option (50ms)
#exercise
    ✓ reverts when not ITM
    ✓ reverts when past expiry
#canExercise
    ✓ can exercise
    ✓ cannot exercise twice
    ✓ cannot exercise after expiry
WBTC CALL ITM
#premium
    ✓ gets premium of option
#purchase
    ✓ reverts when not enough value is passed
    ✓ reverts when buying after expiry
    ✓ reverts when passing unknown underlying
    ✓ creates options on hegic (89ms)
#exerciseProfit
    ✓ reverts when unknown options address passed
    ✓ gets correct exercise profit for an option (80ms)
#exercise
    ✓ exercises options with profit (63ms)
    ✓ redirects exercise profit to recipient (53ms)
    ✓ reverts when past expiry
#canExercise
    ✓ can exercise
    ✓ cannot exercise twice (38ms)
    ✓ cannot exercise after expiry
WBTC CALL OTM
#premium
    ✓ gets premium of option
#purchase
    ✓ reverts when not enough value is passed
    ✓ reverts when buying after expiry
    ✓ reverts when passing unknown underlying
    ✓ creates options on hegic (75ms)
#exerciseProfit
    ✓ reverts when unknown options address passed
    ✓ gets correct exercise profit for an option (82ms)
#exercise
    ✓ reverts when not ITM
    ✓ reverts when past expiry
#canExercise
    ✓ can exercise
    ✓ cannot exercise twice
    ✓ cannot exercise after expiry
WBTC PUT ITM
#premium
    ✓ gets premium of option
#purchase
    ✓ reverts when not enough value is passed
    ✓ reverts when buying after expiry
    ✓ reverts when passing unknown underlying
    ✓ creates options on hegic (84ms)
#exerciseProfit
    ✓ reverts when unknown options address passed
    ✓ gets correct exercise profit for an option (85ms)
#exercise
    ✓ exercises options with profit (139ms)
    ✓ redirects exercise profit to recipient (61ms)
    ✓ reverts when past expiry
#canExercise
    ✓ can exercise
    ✓ cannot exercise twice (60ms)
    ✓ cannot exercise after expiry
WBTC PUT OTM
#premium
    ✓ gets premium of option
#purchase
    ✓ reverts when not enough value is passed
    ✓ reverts when buying after expiry
    ✓ reverts when passing unknown underlying
    ✓ creates options on hegic (84ms)
#exerciseProfit
    ✓ reverts when unknown options address passed (215ms)
    ✓ gets correct exercise profit for an option (80ms)
#exercise
    ✓ reverts when not ITM
    ✓ reverts when past expiry
#canExercise
    ✓ can exercise
    ✓ cannot exercise twice
    ✓ cannot exercise after expiry

UniswapAdapter
#protocolName
    - matches the protocol name
#nonFungible
    - matches the nonFungible bool
buying on UNISWAP
#expectedAmountsOut
    - handles invalid exchange for expectedWbtcOut
    - handles invalid exchange for expectedDiggOut
#buyLp
    - handles invalid exchange using eth
    - handles invalid exchange using btc
    - reverts excessive input amt using eth
    - reverts excessive input amt using btc
    - valid purchase with eth
    - valid purchase with btc
    - purchase with eth using expected amounts out from adapter
    - purchase with btc using expected amountOuts from adapter
buying on UNISWAP
#expectedAmountsOut
    - handles invalid exchange for expectedWbtcOut
    - handles invalid exchange for expectedDiggOut
#buyLp
    - handles invalid exchange using eth
    - handles invalid exchange using btc
    - reverts excessive input amt using eth
    - reverts excessive input amt using btc
    - valid purchase with eth
    - valid purchase with btc
    - purchase with eth using expected amounts out from adapter
```

```
- purchase with wbtc using expected amountOuts from adapter
buying on SUSHISWAP
#expectedAmountsOut
- handles invalid exchange for expectedWbtcOut
- handles invalid exchange for expectedDiggOut
#buyLp
- handles invalid exchange using eth
- handles invalid exchange using wbtc
- reverts excessive input amt using eth
- reverts excessive input amt using wbtc
- valid purchase with eth
- valid purchase with wbtc
- purchase with eth using expected amounts out from adapter
- purchase with wbtc using expected amountOuts from adapter
buying on SUSHISWAP
#expectedAmountsOut
- handles invalid exchange for expectedWbtcOut
- handles invalid exchange for expectedDiggOut
#buyLp
- handles invalid exchange using eth
- handles invalid exchange using wbtc
- reverts excessive input amt using eth
- reverts excessive input amt using wbtc
- valid purchase with eth
- valid purchase with wbtc
- purchase with eth using expected amounts out from adapter
- purchase with wbtc using expected amountOuts from adapter

RibbonFactory
✓ initializes factory correctly
✓ reverts if any account other than owner calls
#setAdapter
✓ sets the adapter
✓ reverts when not owner
#getAdapter
✓ gets the hegic adapter
#adapters
✓ gets the adapters array
#burnGasTokens
✓ cannot burn if not instrument

RibbonThetaVault
Ribbon WBTC Theta Vault (Call)
constructor
✓ reverts when deployed with 0x0 factory
✓ reverts when adapter not set yet (76ms)
✓ reverts when asset is 0x
✓ reverts when decimals is 0
✓ reverts when minimumSupply is 0
✓ sets the correct asset, decimals and minimum supply (133ms)
#initialize
✓ initializes with correct values (133ms)
✓ cannot be initialized twice
✓ reverts when initializing with 0 owner
✓ reverts when initializing with 0 feeRecipient
✓ reverts when initializing with 0 initCap
#name
✓ returns the name
#symbol
✓ returns the symbol
#isPut
✓ returns the correct option type
#delay
✓ returns the delay
#asset
✓ returns the asset
#owner
✓ returns the owner
#setManager
✓ reverts when setting 0x0 as manager
✓ reverts when not owner call
✓ sets the first manager
✓ changes the manager (39ms)
#setFeeRecipient
✓ reverts when setting 0x0 as feeRecipient
✓ reverts when not owner call
✓ changes the fee recipient
#deposit
✓ deposits successfully (41ms)
✓ consumes less than 100k gas in ideal scenario (39ms)
✓ returns the correct number of shares back (55ms)
✓ accounts for the amounts that are locked (180ms)
✓ reverts when no value passed
✓ does not inflate the share tokens on initialization
✓ reverts when minimum shares are not minted
signing an order message
✓ signs an order message
#commitAndClose
✓ reverts when not called with manager
✓ reverts when option is 0x0
✓ reverts when otoken underlying is different from vault's underlying (15581ms)
✓ reverts when otoken collateral is different from vault's asset (15693ms)
✓ reverts when the strike is not USDC (15600ms)
✓ reverts when the option type does not match (15598ms)
✓ reverts when the expiry is before the delay (227ms)
✓ sets the next option and closes existing short (52ms)
✓ should set the next option twice (50ms)
#closeShort
✓ doesnt do anything when no existing short
✓ reverts when closing short before expiry (151ms)
✓ closes the short after expiry (10856ms)
#rollToNextOption
✓ reverts when not called with manager
✓ reverts when delay not passed (38ms)
✓ mints oTokens and deposits collateral into vault (219ms)
✓ reverts when calling before expiry (138ms)
✓ withdraws and roll funds into next option, after expiry ITM (16265ms)
✓ withdraws and roll funds into next option, after expiry OTM (5622ms)
✓ is not able to roll to new option consecutively without setNextOption (109ms)
#emergencyWithdrawFromShort
✓ reverts when not allocated to a short (41ms)
✓ withdraws locked funds by closing short (258ms)
#sellOptions
✓ completes the trade with the counterparty (5264ms)
✓ reverts when not selling option token
✓ reverts when not buying asset token
✓ reverts when sender.wallet is not vault
#assetBalance
✓ returns the free balance, after locking
✓ returns the free balance - locked, if free > locked
#withdrawAmountWithShares
✓ returns the correct withdrawal amount (44ms)
#maxWithdrawAmount
✓ returns the max withdrawable amount accounting for the MINIMUM_SUPPLY
✓ returns the max withdrawable amount
#maxWithdrawableShares
✓ returns the max shares withdrawable of the system
#accountVaultBalance
✓ returns the ETH balance of the account in the vault
#assetAmountToShares
✓ should return the correct number of shares
#withdrawLater
✓ is within the gas budget
✓ rejects a withdrawLater of 0 shares
✓ rejects a scheduled withdrawal when greater than balance
✓ accepts a withdrawLater if less than or equal to balance (42ms)
✓ rejects a withdrawLater if a withdrawal is already scheduled
✓ assets reserved by withdrawLater are not used to short (152ms)
completeScheduledWithdrawal
✓ is within the gas budget (44ms)
✓ rejects a completeScheduledWithdrawal if nothing scheduled
✓ completeScheduledWithdraw behaves as expected for valid scheduled withdraw (313ms)
✓ rejects second attempted completeScheduledWithdraw (274ms)
#withdraw
✓ reverts when withdrawing more than balance (128ms)
✓ should withdraw funds, sending withdrawal fee to feeRecipient (55ms)
✓ should withdraw funds, sending withdrawal fee to feeRecipient if <10% (103ms)
✓ should withdraw funds up to 10% of pool (56ms)
✓ should only withdraw original deposit amount minus fees if vault doesn't expand (71ms)
✓ should withdraw more collateral when the balance increases (65ms)
```



```
    ✓ should revert if not enough shares
    ✓ should be able to withdraw everything from the vault, leaving behind minimum (56ms)
    ✓ should revert when burning past minimum supply
#setCap
    ✓ should revert if not manager
    ✓ should set the new cap
    ✓ should revert when depositing over the cap
#setWithdrawalFee
    ✓ reverts when not manager
    ✓ reverts when withdrawal fee is 0
    ✓ reverts when withdrawal fee set to 30%
    ✓ sets the withdrawal fee
#currentOptionExpiry
    ✓ should return 0 when currentOption not set
#decimals
    ✓ should return 18 for decimals
Ribbon ETH Theta Vault (Call)
constructor
    ✓ reverts when deployed with 0x0 factory (38ms)
    ✓ reverts when adapter not set yet (81ms)
    ✓ reverts when asset is 0x
    ✓ reverts when decimals is 0
    ✓ reverts when minimumSupply is 0
    ✓ sets the correct asset, decimals and minimum supply (151ms)
#initialize
    ✓ initializes with correct values (67ms)
    ✓ cannot be initialized twice
    ✓ reverts when initializing with 0 owner
    ✓ reverts when initializing with 0 feeRecipient
    ✓ reverts when initializing with 0 initCap
#name
    ✓ returns the name
#symbol
    ✓ returns the symbol
#isPut
    ✓ returns the correct option type
#delay
    ✓ returns the delay
#asset
    ✓ returns the asset
#owner
    ✓ returns the owner
#setManager
    ✓ reverts when setting 0x0 as manager
    ✓ reverts when not owner call
    ✓ sets the first manager
    ✓ changes the manager
#setFeeRecipient
    ✓ reverts when setting 0x0 as feeRecipient
    ✓ reverts when not owner call
    ✓ changes the fee recipient
#depositETH
    ✓ deposits successfully
    ✓ consumes less than 100k gas in ideal scenario
    ✓ returns the correct number of shares back (57ms)
    ✓ accounts for the amounts that are locked (175ms)
    ✓ reverts when no value passed
    ✓ does not inflate the share tokens on initialization
    ✓ reverts when minimum shares are not minted
#deposit
    ✓ deposits successfully (68ms)
    ✓ consumes less than 100k gas in ideal scenario
    ✓ returns the correct number of shares back (85ms)
    ✓ accounts for the amounts that are locked (150ms)
    ✓ reverts when no value passed
    ✓ does not inflate the share tokens on initialization
    ✓ reverts when minimum shares are not minted
signing an order message
    ✓ signs an order message
#commitAndClose
    ✓ reverts when not called with manager
    ✓ reverts when option is 0x0
    ✓ reverts when otoken underlying is different from vault's underlying (88ms)
    ✓ reverts when otoken collateral is different from vault's asset (15621ms)
    ✓ reverts when the strike is not USDC (15505ms)
    ✓ reverts when the option type does not match (87ms)
    ✓ reverts when the expiry is before the delay (56ms)
    ✓ sets the next option and closes existing short (39ms)
    ✓ should set the next option twice (44ms)
#closeShort
    ✓ doesnt do anything when no existing short
    ✓ reverts when closing short before expiry (131ms)
    ✓ closes the short after expiry (10776ms)
#rollToNextOption
    ✓ reverts when not called with manager
    ✓ reverts when delay not passed
    ✓ mints oTokens and deposits collateral into vault (131ms)
    ✓ reverts when calling before expiry (150ms)
    ✓ withdraws and roll funds into next option, after expiry ITM (15987ms)
    ✓ withdraws and roll funds into next option, after expiry OTM (5912ms)
    ✓ is not able to roll to new option consecutively without setNextOption (185ms)
#emergencyWithdrawFromShort
    ✓ reverts when not allocated to a short
    ✓ withdraws locked funds by closing short (339ms)
#sellOptions
    ✓ completes the trade with the counterparty (5351ms)
    ✓ reverts when not selling option token
    ✓ reverts when not buying asset token
    ✓ reverts when sender.wallet is not vault
#assetBalance
    ✓ returns the free balance, after locking
    ✓ returns the free balance - locked, if free > locked
#withdrawETH
    ✓ reverts when withdrawing more than balance (123ms)
    ✓ should withdraw funds, sending withdrawal fee to feeRecipient if <10% (57ms)
    ✓ should withdraw funds up to 10% of pool (47ms)
    ✓ should only withdraw original deposit amount minus fees if vault doesn't expand (59ms)
    ✓ should withdraw more collateral when the balance increases (56ms)
    ✓ should revert if not enough shares (39ms)
    ✓ should be able to withdraw everything from the vault, leaving behind minimum (49ms)
    ✓ should revert when burning past minimum supply
#withdrawAmountWithShares
    ✓ returns the correct withdrawal amount (53ms)
#maxWithdrawAmount
    ✓ returns the max withdrawable amount accounting for the MINIMUM_SUPPLY
    ✓ returns the max withdrawable amount
#maxWithdrawableShares
    ✓ returns the max shares withdrawable of the system
#accountVaultBalance
    ✓ returns the ETH balance of the account in the vault
#assetAmountToShares
    ✓ should return the correct number of shares (39ms)
#withdrawLater
    ✓ is within the gas budget
    ✓ rejects a withdrawLater of 0 shares
    ✓ rejects a scheduled withdrawal when greater than balance
    ✓ accepts a withdrawLater if less than or equal to balance (47ms)
    ✓ rejects a withdrawLater if a withdrawal is already scheduled
    ✓ assets reserved by withdrawLater are not used to short (156ms)
completeScheduledWithdrawal
    ✓ is within the gas budget (38ms)
    ✓ rejects a completeScheduledWithdrawal if nothing scheduled
    ✓ completeScheduledWithdraw behaves as expected for valid scheduled withdraw (315ms)
    ✓ rejects second attempted completeScheduledWithdraw (143ms)
#withdraw
    ✓ reverts when withdrawing more than balance (119ms)
    ✓ should withdraw funds, sending withdrawal fee to feeRecipient (71ms)
    ✓ should withdraw funds, sending withdrawal fee to feeRecipient if <10% (79ms)
    ✓ should withdraw funds up to 10% of pool (47ms)
    ✓ should only withdraw original deposit amount minus fees if vault doesn't expand (52ms)
    ✓ should withdraw more collateral when the balance increases (195ms)
    ✓ should revert if not enough shares
    ✓ should be able to withdraw everything from the vault, leaving behind minimum (49ms)
    ✓ should revert when burning past minimum supply
#setCap
    ✓ should revert if not manager
    ✓ should set the new cap
    ✓ should revert when depositing over the cap
#setWithdrawalFee
    ✓ reverts when not manager
```

```
    ✓ reverts when withdrawal fee is 0
    ✓ reverts when withdrawal fee set to 30%
    ✓ sets the withdrawal fee
#currentOptionExpiry
    ✓ should return 0 when currentOption not set
#decimals
    ✓ should return 18 for decimals
Ribbon WBTC Theta Vault (Put)
constructor
    ✓ reverts when deployed with 0x0 factory
    ✓ reverts when adapter not set yet (79ms)
    ✓ reverts when asset is 0x (38ms)
    ✓ reverts when decimals is 0
    ✓ reverts when minimumSupply is 0
    ✓ sets the correct asset, decimals and minimum supply (106ms)
#initialize
    ✓ initializes with correct values
    ✓ cannot be initialized twice
    ✓ reverts when initializing with 0 owner
    ✓ reverts when initializing with 0 feeRecipient
    ✓ reverts when initializing with 0 initCap
#name
    ✓ returns the name
#symbol
    ✓ returns the symbol
#isPut
    ✓ returns the correct option type
#delay
    ✓ returns the delay
#asset
    ✓ returns the asset
#owner
    ✓ returns the owner
#setManager
    ✓ reverts when setting 0x0 as manager
    ✓ reverts when not owner call
    ✓ sets the first manager
    ✓ changes the manager (48ms)
#setFeeRecipient
    ✓ reverts when setting 0x0 as feeRecipient
    ✓ reverts when not owner call
    ✓ changes the fee recipient
#deposit
    ✓ deposits successfully (53ms)
    ✓ consumes less than 100k gas in ideal scenario (75ms)
    ✓ returns the correct number of shares back (63ms)
    ✓ accounts for the amounts that are locked (405ms)
    ✓ reverts when no value passed
    ✓ does not inflate the share tokens on initialization
    ✓ reverts when minimum shares are not minted
signing an order message
    ✓ signs an order message
#commitAndClose
    ✓ reverts when not called with manager
    ✓ reverts when option is 0x0
    ✓ reverts when otoken underlying is different from vault's underlying (15617ms)
    ✓ reverts when otoken collateral is different from vault's asset (15535ms)
    ✓ reverts when the strike is not USDC (15550ms)
    ✓ reverts when the option type does not match (128ms)
    ✓ reverts when the expiry is before the delay (68ms)
    ✓ sets the next option and closes existing short (56ms)
    ✓ should set the next option twice (44ms)
#closeShort
    ✓ doesnt do anything when no existing short
    ✓ reverts when closing short before expiry (165ms)
    ✓ closes the short after expiry (374ms)
#rollToNextOption
    ✓ reverts when not called with manager
    ✓ reverts when delay not passed
    ✓ mints oTokens and deposits collateral into vault (264ms)
    ✓ reverts when calling before expiry (141ms)
    ✓ withdraws and roll funds into next option, after expiry ITM (5743ms)
    ✓ withdraws and roll funds into next option, after expiry OTM (5827ms)
    ✓ is not able to roll to new option consecutively without setNextOption (118ms)
#emergencyWithdrawFromShort
    ✓ reverts when not allocated to a short
    ✓ withdraws locked funds by closing short (250ms)
#sellOptions
    ✓ completes the trade with the counterparty (5224ms)
    ✓ reverts when not selling option token
    ✓ reverts when not buying asset token
    ✓ reverts when sender.wallet is not vault
#assetBalance
    ✓ returns the free balance, after locking
    ✓ returns the free balance - locked, if free > locked (38ms)
#withdrawAmountWithShares
    ✓ returns the correct withdrawal amount (54ms)
#maxWithdrawAmount
    ✓ returns the max withdrawable amount accounting for the MINIMUM_SUPPLY
    ✓ returns the max withdrawable amount
#maxWithdrawableShares
    ✓ returns the max shares withdrawable of the system
#accountVaultBalance
    ✓ returns the ETH balance of the account in the vault
#assetAmountToShares
    ✓ should return the correct number of shares
#withdrawLater
    ✓ is within the gas budget
    ✓ rejects a withdrawLater of 0 shares
    ✓ rejects a scheduled withdrawal when greater than balance
    ✓ accepts a withdrawLater if less than or equal to balance (38ms)
    ✓ rejects a withdrawLater if a withdrawal is already scheduled
    ✓ assets reserved by withdrawLater are not used to short (149ms)
completeScheduledWithdrawal
    ✓ is within the gas budget (41ms)
    ✓ rejects a completeScheduledWithdrawal if nothing scheduled
    ✓ completeScheduledWithdraw behaves as expected for valid scheduled withdraw (189ms)
    ✓ rejects second attempted completeScheduledWithdraw (167ms)
#withdraw
    ✓ reverts when withdrawing more than balance (135ms)
    ✓ should withdraw funds, sending withdrawal fee to feeRecipient (64ms)
    ✓ should withdraw funds, sending withdrawal fee to feeRecipient if <10% (81ms)
    ✓ should withdraw funds up to 10% of pool (101ms)
    ✓ should only withdraw original deposit amount minus fees if vault doesn't expand (102ms)
    ✓ should withdraw more collateral when the balance increases (84ms)
    ✓ should revert if not enough shares
    ✓ should be able to withdraw everything from the vault, leaving behind minimum (55ms)
    ✓ should revert when burning past minimum supply
#setCap
    ✓ should revert if not manager
    ✓ should set the new cap
    ✓ should revert when depositing over the cap
#setWithdrawalFee
    ✓ reverts when not manager
    ✓ reverts when withdrawal fee is 0
    ✓ reverts when withdrawal fee set to 30%
    ✓ sets the withdrawal fee
#currentOptionExpiry
    ✓ should return 0 when currentOption not set
#decimals
    ✓ should return 18 for decimals
Ribbon ETH Theta Vault (Put)
constructor
    ✓ reverts when deployed with 0x0 factory
    ✓ reverts when adapter not set yet (69ms)
    ✓ reverts when asset is 0x
    ✓ reverts when decimals is 0
    ✓ reverts when minimumSupply is 0
    ✓ sets the correct asset, decimals and minimum supply (99ms)
#initialize
    ✓ initializes with correct values (98ms)
    ✓ cannot be initialized twice
    ✓ reverts when initializing with 0 owner
    ✓ reverts when initializing with 0 feeRecipient
    ✓ reverts when initializing with 0 initCap
#name
    ✓ returns the name
#symbol
    ✓ returns the symbol
#isPut
```



```
    ✓ returns the correct option type
#delay
    ✓ returns the delay
#asset
    ✓ returns the asset
#owner
    ✓ returns the owner
#setManager
    ✓ reverts when setting 0x0 as manager
    ✓ reverts when not owner call
    ✓ sets the first manager
    ✓ changes the manager
#setFeeRecipient
    ✓ reverts when setting 0x0 as feeRecipient
    ✓ reverts when not owner call
    ✓ changes the fee recipient
#deposit
    ✓ deposits successfully
    ✓ consumes less than 100k gas in ideal scenario
    ✓ returns the correct number of shares back (112ms)
    ✓ accounts for the amounts that are locked (232ms)
    ✓ reverts when no value passed
    ✓ does not inflate the share tokens on initialization
    ✓ reverts when minimum shares are not minted
signing an order message
    ✓ signs an order message
#commitAndClose
    ✓ reverts when not called with manager
    ✓ reverts when option is 0x0
    ✓ reverts when otoken underlying is different from vault's underlying (15505ms)
    ✓ reverts when otoken collateral is different from vault's asset (15535ms)
    ✓ reverts when the strike is not USDC (15480ms)
    ✓ reverts when the option type does not match (82ms)
    ✓ reverts when the expiry is before the delay (54ms)
    ✓ sets the next option and closes existing short (53ms)
    ✓ should set the next option twice
#closeShort
    ✓ doesnt do anything when no existing short
    ✓ reverts when closing short before expiry (137ms)
    ✓ closes the short after expiry (294ms)
#rollToNextOption
    ✓ reverts when not called with manager
    ✓ reverts when delay not passed (53ms)
    ✓ mints oTokens and deposits collateral into vault (228ms)
    ✓ reverts when calling before expiry (171ms)
    ✓ withdraws and roll funds into next option, after expiry ITM (5967ms)
    ✓ withdraws and roll funds into next option, after expiry OTM (5614ms)
    ✓ is not able to roll to new option consecutively without setNextOption (110ms)
#emergencyWithdrawFromShort
    ✓ reverts when not allocated to a short
    ✓ withdraws locked funds by closing short (313ms)
#sellOptions
    ✓ completes the trade with the counterparty (5355ms)
    ✓ reverts when not selling option token
    ✓ reverts when not buying asset token
    ✓ reverts when sender.wallet is not vault
#assetBalance
    ✓ returns the free balance, after locking
    ✓ returns the free balance - locked, if free > locked (45ms)
#withdrawAmountWithShares
    ✓ returns the correct withdrawal amount (50ms)
#maxWithdrawAmount
    ✓ returns the max withdrawable amount accounting for the MINIMUM_SUPPLY
    ✓ returns the max withdrawable amount (43ms)
#maxWithdrawableShares
    ✓ returns the max shares withdrawable of the system
#accountVaultBalance
    ✓ returns the ETH balance of the account in the vault
#assetAmountToShares
    ✓ should return the correct number of shares (50ms)
#withdrawLater
    ✓ is within the gas budget
    ✓ rejects a withdrawLater of 0 shares
    ✓ rejects a scheduled withdrawal when greater than balance
    ✓ accepts a withdrawLater if less than or equal to balance (110ms)
    ✓ rejects a withdrawLater if a withdrawal is already scheduled
    ✓ assets reserved by withdrawLater are not used to short (154ms)
completeScheduledWithdrawal
    ✓ is within the gas budget (45ms)
    ✓ rejects a completeScheduledWithdrawal if nothing scheduled
    ✓ completeScheduledWithdraw behaves as expected for valid scheduled withdraw (210ms)
    ✓ rejects second attempted completeScheduledWithdraw (160ms)
#withdraw
    ✓ reverts when withdrawing more than balance (127ms)
    ✓ should withdraw funds, sending withdrawal fee to feeRecipient (51ms)
    ✓ should withdraw funds, sending withdrawal fee to feeRecipient if <10% (140ms)
    ✓ should withdraw funds up to 10% of pool (52ms)
    ✓ should only withdraw original deposit amount minus fees if vault doesn't expand (72ms)
    ✓ should withdraw more collateral when the balance increases (65ms)
    ✓ should revert if not enough shares
    ✓ should be able to withdraw everything from the vault, leaving behind minimum (47ms)
    ✓ should revert when burning past minimum supply
#setCap
    ✓ should revert if not manager
    ✓ should set the new cap
    ✓ should revert when depositing over the cap
#setWithdrawalFee
    ✓ reverts when not manager
    ✓ reverts when withdrawal fee is 0
    ✓ reverts when withdrawal fee set to 30%
    ✓ sets the withdrawal fee (50ms)
#currentOptionExpiry
    ✓ should return 0 when currentOption not set
#decimals
    ✓ should return 18 for decimals

StakedPut
constructor
    ✓ reverts when deployed with 0x0 factory
#initialize
    ✓ initializes with correct values
    ✓ cannot be initialized twice
#name
    ✓ returns the name
Hegic ITM Put, SUSHISWAP LP, ETH PMT
    ✓ test oracle (61ms)
    ✓ test option buy (238ms)
    ✓ reverts instrument buy on invalid payment token (57ms)
gas used 961083
sent a value of 111885390631825504818
user sushi lp balance after trade of 703593648
wbtc size 5888933645238000000000
option size is 5888933645238
    ✓ test instrument buy (376ms)
Hegic ITM Put, SUSHISWAP LP, ETH PMT SMALL AMT
    ✓ test oracle (49ms)
    ✓ test option buy (173ms)
    ✓ reverts instrument buy on invalid payment token (45ms)
gas used 960951
sent a value of 1118315444173253
user sushi lp balance after trade of 7055
wbtc size 5888933645238000000000
option size is 5888933645238
    ✓ test instrument buy (407ms)
Hegic OTM Put, SUSHISWAP LP, ETH PMT SMALL AMT
    ✓ test oracle (57ms)
    ✓ test option buy (198ms)
    ✓ reverts instrument buy on invalid payment token (48ms)
gas used 960951
sent a value of 1118315444173253
user sushi lp balance after trade of 7055
wbtc size 5888933645238000000000
option size is 5888933645238
    ✓ test instrument buy (344ms)

RibbonOptionsVault Upgrade
Upgrade from 4ee578b96aefa663458ec8f871732fb21fa0ceb9
    ✓ performs a sanity check of the storage slots before upgrade (344ms)
    ✓ performs upgrade and storage is intact (484ms)
```



```
VaultRegistry
#constructor
    ✓ initializes the owner correctly
#registerFreeWithdrawal
    ✓ registers free withdrawal
    ✓ reverts when not owner
#revokeFreeWithdrawal
    ✓ revokes free withdrawal
    ✓ reverts when not owner
#registerCrossTrade
    ✓ registers cross trade
    ✓ reverts when not owner
#revokeCrossTrade
    ✓ revokes free withdrawal
    ✓ reverts when not owner
```

Code Coverage

Code coverage may be skewed due to 4 tests that only failed during coverage. We recommend investigating the coverage issues and ensuring high coverage on all contracts.

```

1) RibbonThetaVault
  Ribbon WBTC Theta Vault (Put)
    #deposit
      consumes less than 100k gas in ideal scenario:
    AssertionError: expected 101006 to be at most 100000
    + expected - actual
    -101006
    +100000
    at Context.<anonymous> (test/RibbonThetaVault.js:906:16)
    at runMicrotasks (<anonymous>)
    at processTicksAndRejections (internal/process/task_queues.js:93:5)
2) RibbonThetaVault
  Ribbon WBTC Theta Vault (Put)
    completeScheduledWithdrawal
      is within the gas budget:
    AssertionError: expected 83424 to be at most 80000
    + expected - actual
    -83424
    +80000
    at Context.<anonymous> (test/RibbonThetaVault.js:2563:16)
    at runMicrotasks (<anonymous>)
    at processTicksAndRejections (internal/process/task_queues.js:93:5)
3) RibbonThetaVault
  Ribbon ETH Theta Vault (Put)
    #deposit
      consumes less than 100k gas in ideal scenario:
    AssertionError: expected 101006 to be at most 100000
    + expected - actual
    -101006
    +100000
    at Context.<anonymous> (test/RibbonThetaVault.js:906:16)
    at runMicrotasks (<anonymous>)
    at processTicksAndRejections (internal/process/task_queues.js:93:5)
4) RibbonThetaVault
  Ribbon ETH Theta Vault (Put)
    completeScheduledWithdrawal
      is within the gas budget:
    AssertionError: expected 83424 to be at most 80000
    + expected - actual
    -83424
    +80000
    at Context.<anonymous> (test/RibbonThetaVault.js:2563:16)
    at runMicrotasks (<anonymous>)
    at processTicksAndRejections (internal/process/task_queues.js:93:5)

```

File	% StmtS	% Branch	% Funcs	% Lines	Uncovered Lines
contracts/	25	25	36.36	25	
RibbonFactory.sol	35	25	57.14	35	... ,98,103,104
VaultRegistry.sol	0	100	0	0	... 58,59,71,72
contracts/adapters/	45.25	38.04	30.95	45.39	
AmmAdapter.sol	0	0	0	0	... 37,46,48,52
CharmAdapter.sol	5.06	6.67	6.25	5.06	... 395,402,403
GammaAdapter.sol	95.3	60.29	83.33	95.27	... 211,446,654
HegicAdapter.sol	52.21	50	52.63	52.29	... 411,420,424
IAmmAdapter.sol	100	100	100	100	
IProtocolAdapter.sol	100	100	100	100	
ProtocolAdapter.sol	0	0	0	0	... 211,213,217
UniswapAdapter.sol	0	0	0	0	... 261,262,263
contracts/archive/	0	0	0	0	
RibbonVolatility.sol	0	0	0	0	... 496,498,500
contracts/experimental/	0	0	0	0	
StakedPut.sol	0	0	0	0	... 298,301,302
contracts/instruments/	0	0	0	0	
RibbonThetaVault.sol	0	0	0	0	... 652,659,660
All files	23.79	18.98	20	23.91	

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

5748babc603e8ac7c4925476f1e1669a48e94b7b3ccfe656a2cf1afd8a1e4acd ./contracts/RibbonFactory.sol

bd12930ccfef4c49d4c90d69c5c9617d9a2dac6368d605a4bfa893fc6427d881 ./contracts/storage/OptionsVaultStorage.sol

ef55b5496b88b7451bd2780dcba832738ab1ab31d089a2e259d0cb5afbb016aa ./contracts/lib/DSMath.sol

8d3fd57f93c1890dbdbd12b5a3922455abf83a8c5f430e9f0a5d399ce9564763 ./contracts/lib/Ownable.sol

5b40ade1e7101073733c2869ec9122f94edd735e6406a413d316165001da358a ./contracts/lib/upgrades/AdminUpgradeabilityProxy.sol

a084222f4417bbaf05c925b7ee68a2e2e7af0dd3aa7dd4a17d46d403f170c7b5 ./contracts/lib/upgrades/Initializable.sol

12849bf08d36412c4da0e2cdd0de457a570d41e55e398fc3e6f5ccf4b1d96711 ./contracts/lib/upgrades/Proxy.sol

68442066e61fb8b5c7c0bb6bbd2d830f3fc384ced2d6a3f8d64cd165b3c52806 ./contracts/lib/upgrades/ProxyAdmin.sol

63a805be5dfa97dcf98368e88998732dfea9c23b658b5d853fb224c48be28fd7 ./contracts/lib/upgrades/UpgradeabilityProxy.sol

7e03dcd80a9ef7b196cc056e1e064e578770b74646a4f0b2e2f93efedf33fcae ./contracts/interfaces/GammaInterface.sol

8a66d9bc8c3b253556c796ad4b754ad62cd4b8cbec2e06379a95516c54c837cb ./contracts/interfaces/HegicInterface.sol

57cc821d25ed17ad4599d60d17281f315d7a5beae4df5bd755427092dc7fe7eb ./contracts/interfaces/IChiToken.sol

76ceb27f1fc5b819884509c302b4d743b0bc8f5475c6227a9c8bee974b3ac48d ./contracts/interfaces/InstrumentInterface.sol

059ee15a8e6f14ecb01532e7d4f56d3caeb6b2fc81675b701583cc5b6fea30af ./contracts/interfaces/IRibbonFactory.sol

acb8833f7086e33af4e10b8e66fe721706f8d6bdd8644029e4d517a1d11f49ce ./contracts/interfaces/ISwap.sol

077d1d2796282060743d357a7880554a01a49a5b992219a3c78db32a5d1e7002 ./contracts/interfaces/ISwapPair.sol

8029566c69986499ec31dd6c3a62c980a4735a852210b63a36eacb1be8e9efcd ./contracts/interfaces/IUniswapV2Pair.sol

26386698907242f683be001d62fdfc3a28b68a5a0024d9bbd24882f2ca79e4c0 ./contracts/interfaces/IUniswapV2Router.sol

d04b5599eb6f045cd49de84cf4331c089890f913e7770c069fcf6c8f1f344ec6 ./contracts/interfaces/IWETH.sol

bde88ae0d69fa1ec420276dbe30426e3d18727d5ba3762b494156824f0721ad9 ./contracts/interfaces/IZeroExExchange.sol

4610780d9100852512f2b6fb92ff50ac3e4f2580f6ad65b319c9515826980402 ./contracts/instruments/RibbonThetaVault.sol

c47efd07ffc78a5e881a84c0ccbc8a1f625019b31d7272477dabc9986f3c0187 ./contracts/adapters/GammaAdapter.sol

73b64b5748af28a8cbfd80773abfb770db34bdfa95418ad1619494281e7418bc ./contracts/adapters/IProtocolAdapter.sol

3104e0d17b2252417c08fa9a806c4d1fe517f1646e61d64572685acbc45ceff1 ./contracts/adapters/ProtocolAdapter.sol

Tests

ee8a0681e1a1d7d1c84256daed9815c0a13609089e354664972868111e2dcd68 ./test/helper.js

fde0f0e2fe66db018b1b61afe337a6665d515e169a57248e9c51822b1e5e325e ./test/RibbonFactory.js

d558d277ac66fc7cac48dcfe0eb47be33134cb5903f2a182a27229ea38ab26d1 ./test/RibbonThetaVault.js

aff5ec137009ab18b6b0456af3b01b3aca67589901130e4f507eb2ce58da6afc ./test/StakedPut.js

14da16c7119cfe959017f38b764cad630f1fc1e8e1399f1f601a2af14e50f0a3 ./test/VaultRegistry.js

2efff750ca6e0a59d768ced59be59f661b1c9db2ea3bcc805ed1caeff6c198df ./test/upgrades/RibbonThetaVault.js

5c71419d27fc2800f57e71a5edc4247dc80009a09f21ea56007809a249bdd533 ./test/helpers/time.js

e21121c33a21c168b4e8f950972192df903f7e85dac11f9d13fcd6c5d5e71823 ./test/helpers/utils.js

96345873c0fcba47538494ece321d2b84040abe21f97b36a4eecfce22adf304c ./test/adapters/GammaAdapter.js

0541978fa68e5e62c76ba1ef449e52a69ba5703593c8c67c9c98ec11cd779ea7 ./test/adapters/HegicAdapter.js

13de038d7b9059fd5277658751be94ae666d29b007bb315fe19f9d5ca7256d03 ./test/adapters/UniswapAdapter.js

Changelog

- 2021-05-25 - Initial report

About Quantstamp

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

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

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

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

Timeliness of content

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

Notice of confidentiality

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

Links to other websites

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

Disclaimer

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