Quantstamp Security Assessment Certificate

December 14th 2020 — Quantstamp Verified

SirenMarkets

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

Executive Summary

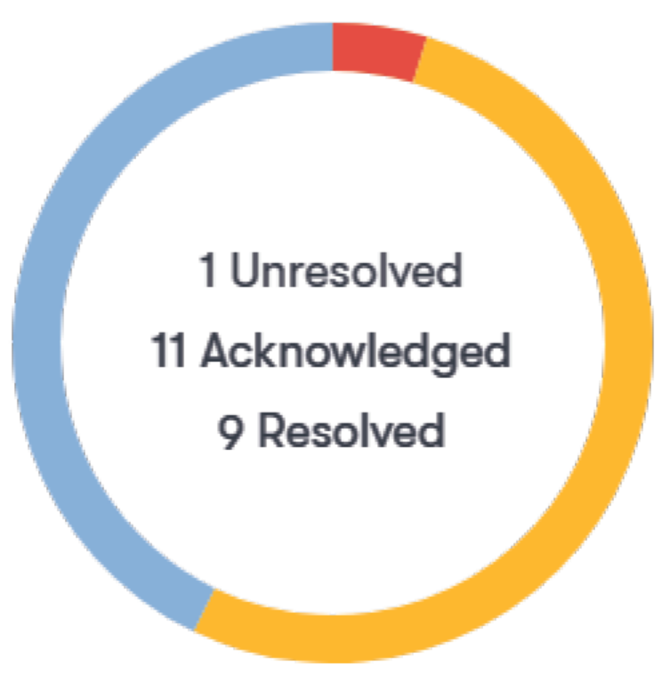
Type	Automated Market Maker (AMM) and Goverance
Auditors	Poming Lee, Research Engineer Fayçal Lalidji, Security Auditor Jake Goh Si Yuan, Research Engineer
Timeline	2020-11-02 through 2020-12-11
EVM	Muir Glacier
Languages	Solidity, Javascript
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	README.md

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.

Goals

- Do functions have proper access control logic?
- Are there centralized components of the system which users should be aware?
- Do the contracts adhere to best practices?

Total Issues	**21**	(9 Resolved)
High Risk Issues	**2**	(1 Resolved)
Medium Risk Issues	**5**	(1 Resolved)
Low Risk Issues	**4**	(2 Resolved)
Informational Risk Issues	**9**	(4 Resolved)
Undetermined Risk Issues	**1**	(1 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

Quantstamp has performed a security audit of the Siren Market project. During auditing, we found eleven potential issues of various levels of severity: one high-severity issue, two medium-severity issues, seven informational-level findings, and one undermined finding. We also made ten best practices recommendations. Overall, the code comment is good for this project. The Solidity Coverage does not work due to compilation error of the solc compiler on the solidity-coverage-modifed contracts. We strongly recommend for Siren team to find a way to fix this and obtain a code coverage report that states that all the code coverage values are at least 90% before go live, to reduce the potential risk of having functional bugs in the code.
** 2020-11-17 update **: during this first reaudit, Siren team has either brought the status of findings into fixed or acknowledged. In addition, we found nine potential issues of various levels of severity: one high-severity issue, three medium-severity issues, three low-severity issues, and two informational-level findings. We also made eight best practices recommendations. Furthermore, it is worth noting that there is still no coverage report for this project.
** 2020-12-11 update **: during this first reaudit, Siren team has either brought the status of findings into fixed or acknowledged. A new `staking` directory was added to this commit and all the files in it were not audited, except `VestingVault.sol`. We found one potential low-severity issue in the file. It is worth noting that there is still no coverage report for this project.

ID	Description	Severity	Status
QSP-1	Withdraw Open Capital	High	Acknowledged
QSP-2	Update Implementation of Market Contract Without Having a Permission	High	Fixed
QSP-3	Missing Input Check When Changing Admin	Medium	Fixed
QSP-4	Market Self Destruct Without Returning Funds	Medium	Acknowledged
QSP-5	Potential Error in Calculating `unclaimedPayment`	Medium	Acknowledged
QSP-6	Possible Stale Oracle Price	Medium	Acknowledged
QSP-7	Possible Unexpected Revert Due to Low Colateral Balance in `MinterAmm.sol`	Medium	Acknowledged
QSP-8	Gas Concerns Related to Unrestrained Number of Markets	Low	Acknowledged
QSP-9	Potential Precision Lose Due To the Order Of Computation	Low	Fixed
QSP-10	Critical Logic May Not be Implemented Based on Specification	Low	Fixed
QSP-11	Cliff Period Logic	Low	Unresolved
QSP-12	Missing Input Sanitization	Informational	Fixed
QSP-13	Privileged Roles	Informational	Acknowledged
QSP-14	`delegateBySig()` Should Validate the `v` and `s` Parameters	Informational	Fixed
QSP-15	Potential Greedy Behavior of The Contract to the Last User to Withdraw	Informational	Fixed
QSP-16	Unlocked Pragma	Informational	Fixed
QSP-17	Centralization of Voting Power With Initial Total Supply of `SRN` All Premined For Vote	Informational	Acknowledged
QSP-18	Contract to Lose Functionalities After `block.number > uint32(-1)`	Informational	Acknowledged
QSP-19	Missing Input Check For `_amm` Before Creating a New Market	Informational	Acknowledged
QSP-20	External Price Oracle is Used so Further Attention is Required	Informational	Acknowledged
QSP-21	Integer Overflow / Underflow	? 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:

- Truffle v5.1.33
- SolidityCoverage v0.7.11
- Mythril v0.22.10
- Slither v0.6.12

Steps taken to run the tools:

1. Installed Truffle: `npm install -g truffle`
2. Installed the solidity-coverage tool (within the project's root directory): `npm install --save-dev solidity-coverage`
3. Ran the coverage tool from the project's root directory: `./node_modules/.bin/solidity-coverage`
4. Installed the Mythril tool from Pypi: `pip3 install mythril`
5. Ran the Mythril tool on each contract: `myth a path/to/contract`
6. Installed the Slither tool: `pip install slither-analyzer`
7. Run Slither from the project directory: `slither .s`

Findings

QSP-1 Withdraw Open Capital

Severity: High Risk

Status: Acknowledged

Description: When the liquidity providers withdraw their collateral from `MinterAmm` using `withdrawCapitalOpen`, their remaining balance of either `wToken` or `bToken` is not sent back. As an example, if a liquidity provider owns 50% of a pool and the pool holds 10 `bToken` and 5 `wToken`, the liquidity providershares will be equal to 5 `bToken` and 2.5 `wToken`. `collateralToClose` will be equal to the smallest value of `wToken` or `bToken` meaning that the liquidity provider will receive 2.5 as collateral token, `closePosition` function member of `Market` contract will burn

2.5 for each `bToken` and `wToken` and the reamining 2.5 `bToken` that are supposed to be owned by the liquidity provider are not given back to the user.And the consequences will be: 1) the liquidity provider loses part of his holdings (either `bToken` or `wToken`), and 2) an arbitrage opportunity is created since executing `withdrawCapitalOpen` will unbalance the AMM pool assets in most cases.
** 2020-11-17 update **: SirenMarket team confirmed that this was intentional by design to penalize an LP for withdrawing before the option expiration.

Recommendation: Send the remaining balance of either `wToken` or `bToken` to the liquidity provider. Otherwise, please make sure if this is intended

QSP-2 Update Implementation of Market Contract Without Having a Permission

Severity: High Risk

Status: Fixed

Description: Function `updateImplementation` in `contracts\market\Market.sol` does not implement a permission control.

Recommendation: Implement some form of validation or access control.

QSP-3 Missing Input Check When Changing Admin

Severity: Medium Risk

Status: Fixed

Description: In `contracts\governance\GovernorAlpha.sol`, function `__queueSetTimelockPendingAdmin` should check if the address `newPendingAdmin` is not `0x0`.

QSP-4 Market Self Destruct Without Returning Funds

Severity: Medium Risk

Status: Acknowledged

Description: Self destructing the `contracts\market\Market.sol` contract using `selfDestructMarket` without withdrawing the remaining amount of collateral token or payment token will lead the asset to be frozen.
** 2020-11-17 update **: tokens left will be sent to the contract owner now after the contract is destroyed. Siren team is recommended to keep good track on the ownership data of all these tokens left in the contract before the the contract is destroyed in order to return them to token owners.

Recommendation: Leave the contract open for all users if they did not claim their assets in time, or add a asset returning mechanism before calling function `selfdestruct`.

QSP-5 Potential Error in Calculating `unclaimedPayment`

Severity: Medium Risk

Status: Acknowledged

Description: In function `getTotalPoolValue` in `contracts\amm\MinterAmm.sol`, the formula of calculating `unclaimedPayment` is based on a hidden assumption of that the `paymentToken` is always a stable coin which its value is pegged to 1USD. If the `paymentToken` is not a stable coin the calculation result will be incorrect. Moreover, if it was intended to be a stable coin only, the developers might need to take into consideration that even stable coins' USD price might drift from their target price from time to time.
** 2020-12-11 **: Siren team documented this assumption in the code, and stated that this assumption will be checked whenever creating a new AMM.

QSP-6 Possible Stale Oracle Price

Severity: Medium Risk

Status: Acknowledged

Description: `MinterAmm.getCurrentCollateralPrice` calls `AggregatorV3Interface.getLatestPrice`. However, `getLatestPrice` does not take into consideration if the price is stale, meaning that the latest round returned can be obsolete since Chainlink aggregators rely on external nodes to be updated once a request is submitted by one of the sponsors.
** 2020-12-11 **: Siren team documented this assumption in the code and stated that they will add logic which returns an error if the price is stale by a certain error tolerance, in the future.

Recommendation: The value returned by Chainlink aggregator must be validated, both `startedAt` and `updatedAt` should be verified correctly.

QSP-7 Possible Unexpected Revert Due to Low Colateral Balance in `MinterAmm.sol`

Severity: Medium Risk

Status: Acknowledged

Description: The two lines of code in function `_sellOrWithdrawActiveTokens` in `MinterAmm.sol`, that are: `collateralLeft = collateralLeft.sub(collateralAmountB);` `collateralLeft = collateralLeft.sub(collateralAmountW);` might revert unexpectedly when uint256 `collateralLeft` pass into the function is too small (which is likely to happen).
** 2020-12-11 **: Siren team documented this in the code and stated that they will handle this error manually when it happened.

Recommendation: Modify the current implementation of code to avoid this.

QSP-8 Gas Concerns Related to Unrestrained Number of Markets

Severity: Low Risk

Status: Acknowledged

Description: For multiple functions in `MinterAmm`, the registered markets list in `MarketsRegistry` is iterated over. Depending on the list lengths and assuming that new markets for the same pair will be added at least every time a market will expire, the gas consumption will grow since added markets are not deleted once expired. Transactions might throw for out of gas or block gas limit.
** 2020-12-11 **: Siren team acknowledged that this is a problem and will add code in the near future for manually pruning markets from the multiple markets arrays, which will restrain the number of markets.

Recommendation: * Run a gas analysis to always be aware of the maximum number of acceptable active and expired markets.

- Delete closed markets from the array in `MarketsRegistry.sol` when self destruct is called.

QSP-9 Potential Precision Lose Due To the Order Of Computation

Severity: Low Risk

Status: Fixed

Description: On `L224`-`L226` in `contracts\amm\MinterAmm.sol`, it is suggested to perform multiplication before the division.

QSP-10 Critical Logic May Not be Implemented Based on Specification

Severity: Low Risk

Status: Fixed

Description: The contract `MinterAmm.sol` uses the function `calcPrice` to determine the price of the `bToken` through the Black-Scholes method. The only documentation is the code comment that states that `Formula: 0.4 * ImplVol * sqrt(timeUntilExpiry) * currentPrice / strike`on `L768`. However, the implementation logic `L776`-`L787` does not include the `0.4` as expected from the described formula, and the parts regarding `intrinsic` part is hard to be verified without precise specification.

Recommendation: Rectify the issue about the inconsistency between specification in documentation and implementation, and explain better what is happening in this logic.

QSP-11 Cliff Period Logic

Severity: Low Risk

Status: Unresolved

Description: In `staking\VestingVault.sol`: the global `startTime` is used to all the introduced grants to calculate the vesting cliff period. A grant that is added after the `startTime` will be able to start withdrawing its vested tokens if the cliff period has passed. This issue is due to the grant creation time not considered when calculating the vested amounts.

Recommendation: Either do not allow grant creation through `addTokenGrant` after `startTime`, or take into account the grant creation timestamp to calculate if the vesting cliff period has passed.

QSP-12 Missing Input Sanitization

Severity: Informational

Status: Fixed

Description: All or part of the input parameters of the listed functions below are not validated, addresses should be checked to be valid depending each use case, fees must be verified to be lowed than their respective denominator and not higher than a certain threshold to ensure user trust, etc ...

- `Market.initialize`
- `MarketsRegistry.initialize`
- `MarketsRegistry.updateTokenImplementation`
- `MarketsRegistry.updateMarketImplementation`
- `MarketsRegistry.updateLpTokenImplementation`
- `MarketsRegistry.updateAmmImplementation`
- `MarketsRegistry.selfDestructMarket`
- `GovernorAlpha.constructor`
- `MinterAmm.initialize`

Recommendation: Add a `require` statement that checks that the value of the `priceOracle_` is different from `0x0`.

QSP-13 Privileged Roles

Severity: Informational

Status: Acknowledged

Description: A. In `MarketsRegistry.sol`, the admin of the contract `contracts\market\MarketsRegistry.sol` has: 1) full control of the listing of markets, 2) can modify the code implementation of each `Market` through `updateMarketsRegistryImplementation()` at any time, and 3) can destroy a market contract by calling the function `selfDestructMarket()`. B. In `GovernorAlpha.sol`, the `guardian` can call function `__queueSetTimelockPendingAdmin` and `__executeSetTimelockPendingAdmin` to change the admin of the `contracts\governance\GovernorAlpha.sol` at anytime without calling for a vote. If `__queueSetTimelockPendingAdmin` is used, there will be an `eta` that is supposed to be further than the current time + the `delay` set in `contracts\governance\Timelock.sol`. Recommendation: make this information explicit to the end users.

Recommendation: These privileged operations and their potential consequences should be clearly communicated to (non-technical) end-users via publicly available documentation.

QSP-14 `delegateBySig()` Should Validate the `v` and `s` Parameters

Severity: Informational

Status: Fixed

Description: For `contracts\governance\SirenToken.sol`, `delegateBySig()` should validate the `v` and `s` parameters as in `ECDSA.sol` (See: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/2bb06b1af4d57cf47c700992b327a08bebf64879/contracts/cryptography/ECDSA.sol#L46).

QSP-15 Potential Greedy Behavior of The Contract to the Last User to Withdraw

Severity: *Informational*

Status: Fixed

Description: (1) In function `claimCollateral` of `contracts\market\Market.sol`, when `collateralToken` is almost empty, the transaction sent by the last user who try to withdraw asset through calling function `claimCollateral` might be reverted on `L393` due to insufficient funds, caused by performing calculations that involve rounding in previous transactions. (2) Similar to (1), it also happens in the same function for `paymentToken` on `L426`.

Recommendation: When encountering insufficient funds condition, should directly send back all the funds that still hold by the contract to the user.

QSP-16 Unlocked Pragma

Severity: *Informational*

Status: Fixed

File(s) affected: `contracts\governance\amm\MinterAmm.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-17 Centralization of Voting Power With Initial Total Supply of `SRN` All Premined For Vote

Severity: *Informational*

Status: Acknowledged

Description: In `contracts\governance\SirenToken.sol`, on `L64` all the supply will initially go to a single account and there is no restriction to that account in regard to its' voting power.

QSP-18 Contract to Lose Functionalities After `block.number > uint32(-1)`

Severity: *Informational*

Status: Acknowledged

Description: In `contracts\governance\SirenToken.sol`, see `L263` and function `_writeCheckpoint` for more. They are all limited by the `safe32` function.
** 2020-11-17 update **: Siren team confirmed that the contract will lose its functionalities after `block.number > uint32(-1)` and could address it in the future.

QSP-19 Missing Input Check For `_amm` Before Creating a New Market

Severity: *Informational*

Status: Acknowledged

Description: In `contracts\market\MarketsRegistry.sol`, the function `createMarket` should check if the address `_amm` is not `0x0`.
** 2020-12-11 **: Siren team decided that there is no need to modify the code since this function is protected by an `onlyOwner` modifier and an honest user would not make this mistake.

QSP-20 External Price Oracle is Used so Further Attention is Required

Severity: *Informational*

Status: Acknowledged

Description: External price oracle has been used for hacking for a while. It has been proved that the price data of a decentralized exchange can be easily manipulated.
** 2020-12-11 **: Siren team decided to further enhance the security level of this part in the future.

Recommendation: Please make sure that the price oracle used in this project is not solely based on decentralized exchanges' price data. Could refer to: https://samczsun.com/so-you-want-to-use-a-price-oracle/ for more information.

QSP-21 Integer Overflow / Underflow

Severity: *Undetermined*

Status: Fixed

Description: Integer overflow/underflow occur when an integer hits its bit-size limit. Every integer has a set range; when that range is passed, the value loops back around. A clock is a good analogy: at 11:59, the minute hand goes to 0, not 60, because 59 is the largest possible minute. Integer overflow and underflow may cause many unexpected kinds of behavior and was the core reason for the `batchOverflow` attack. Here's an example with `uint8` variables, meaning unsigned integers with a range of `0..255`. `function under_over_flow() public { uint8 num_players = 0; num_players = num_players - 1; // 0 - 1 now equals 255! if (num_players == 255) { emit LogUnderflow(); // underflow occurred } uint8 jackpot = 255; jackpot = jackpot + 1; // 255 + 1 now equals 0! if (jackpot == 0) { emit LogOverflow(); // overflow occurred } }`
In particular, the `L278`, `L290`, `L322`, `L369`-`L370`, `L402`-`L403` in `contracts\market\Market.sol`, `SafeMath` was not enforced for arithmetic operations. This would greatly increase the chances of having underflow/overflow issues and lead to unexpected results.

Recommendation: Consider using the `SafeMath` library for all of the arithmetic operations

Automated Analyses

Mythril

Mythril reported no issues.

Slither

- Slither warns of several potential reentrancy issues, however as the associated external calls were to trusted contracts (either Idle contracts or underlying protocols), we classified these as false positives.
- Slither detects that several contracts allow anyone to destruct the contracts, after examining it QuantStamp considers it to be false positives.

Code Documentation

The code is generally well-documented. We suggest several improvements:

1. `contracts\amm\MinterAmm.sol`: `L60-L68` LpTokens -> LPTokens
2. `contracts\amm\MinterAmm.sol`: `L352`, "Send wTokens out 'out'"?
3. `contracts\amm\MinterAmm.sol`: `L30` subracted -> subtracted
4. `contracts\market\Market.sol`: `L14-L16` the document talks about 2 states for market and yet `IMarket` describes 3 states, which is inconsistent.
5. `contracts\market\MarketsRegistry.sol`:L135` updateAmmImplementation -> updateAMMImplentation
6. `contracts\market\MarketsRegistry.sol`:L149` updateLpTokenImplementation -> updateLPTokenImplementation
7. `contracts\market\MarketsRegistry.sol`:L163` updateAmmFees -> updateAMMFees
8. `contracts\market\MarketsRegistry.sol`:L147` AMM -> LP
9. `contracts\governance\Timelock.sol`: `L27-L38` error message should be `exceed or equal`
10. `contracts\governance\Timelock.sol`: `L29` error message should be `Timelock::constructor`11. The AMM mathematical formulation used to calculate the input and output amounts is not descrubed anywhere therefore we cannot prove its correctness, A more detaild mathematical development must be provided to guarentee that the following functions are correct: `MinterAmm.bTokenGetCollateralOut`, `MinterAmm.bTokenGetCollateralIn`, `MinterAmm.wTokenGetCollateralOut`, `MinterAmm.wTokenGetCollateralIn`.

Adherence to Best Practices

The code does not fully adhere to best practices. In particular:

- For the function `provideCollateral` in `contracts\amm\MinterAmm.sol`, `L217-L241` would make the ratio between bToken and wToken more imbalanced. Please make sure if this is intended.
- On `L194-L198` in `contracts\amm\MinterAmm.sol`, if `_wTokenRefund` is specified, `_bTokenRefund` will be ignored. Please make sure if this is intended.
- The `priceRatio` in `contracts\market\Market.sol` will never change, please make sure if this is intended.
- `contracts\market\Market.sol`: on `L224`, `180` days is explicitly stated in `state()` function. This could be set as a public constant state variable that can be viewed instead for clarity (or as a variable, for upgradeability purposes).
- `contracts\market\MarketsRegistry.sol`: `L275` unused state variable __gap.
- `contracts\governance\Timelock.sol`: function in `L107` is unneccessary, can use `block.timestamp` directly instead.
- European and American style of option contracts have different rules, in `contracts\market\Market.sol` and `contracts\market\MarketsRegistry.sol` it appears that currently these different rules are not being implemented, please make sure this intended.
- `contracts\token\LPToken.sol`: `LPToken.sendDistributionFunds` is allowed to be called by any address, it should be authorised for its respective `MinterAmm` only to avoid third party or users mistakes.
- The definition of the strike price ratio in the https://github.com/sirenmarkets/app/blob/master/packages/contracts/README.md#strike-price-ratio is confusing, the ratio has to be above or below 1 regardless if it is a put or a call option. As an example, a call option strike price ratio can be either lower or higher depending on the relation between its strike asset price and underlying asset price. When creating a `Market` contract the option type should be specified to either put or call option, since the strike price ratio does not determine the option type as described. Please make sure if this is intended.
- Function `withdrawCapitalExpired` and `withdrawCapitalOpen` in `contracts\amm\MinterAmm.sol` do not behave in a similar way. For example function `withdrawCapitalOpen` will return the collateral token and `withdrawCapitalExpired` will return the `wToken`, then it is up to the LP to withdraw his collateral token from the `Market` contract. Consider align the behavior to make it looks more consistent from end users' perspective.

** 2020-11-17 newly added**:

- For the function `provideCollateral` in `contracts\amm\MinterAmm.sol`, `L217-L241` would make the ratio between bToken and wToken more imbalanced. Please make sure if this is intended.
- On `L194-L198` in `contracts\amm\MinterAmm.sol`, if `_wTokenRefund` is specified, `_bTokenRefund` will be ignored. Please make sure if this is intended.
- The `priceRatio` in `contracts\market\Market.sol` will never change, please make sure if this is intended.
- On `L759` in `contracts\amm\MinterAmm.sol`, when a market expires, this function will always revert instead of returning a zero. Please confirm if this is intended.
- In `contracts\amm\MinterAmm.sol`, the function `initialize` should check that `_paymentToken` != `_collateralToken`. It could lead to unexpected results in the function `getTotalPoolValue` if they are the same.
- `contracts\amm\MinterAmm.sol`: note sure what `L858` is for.
- `contracts\amm\MinterAmm.sol`: `L175` can move magic number to constant state variable.
- `contracts\amm\MinterAmm.sol`: `L249` lpToken names are packed in the form `LP-WBTC-USDC` which can be confused as a triplet instead of a pair. Consider using a different `--` to distinguish instead.

Test Results

Test Suite Results

All tests have passed.

```
Compiling your contracts...
===========================
> Compiling ./contracts/amm/MinterAmm.sol
> Compiling ./contracts/libraries/Math.sol
> Compiling ./contracts/staking/IERC20.sol
> Compiling ./contracts/staking/RewardsDistribution.sol
> Compiling ./contracts/test/TestUpgradeableAmm.sol
> Compilation warnings encountered:

    /root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/amm/ISirenTradeAMM.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consid
er adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informat
ion.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/amm/InitializeableAmm.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consid
er adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informat
ion.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/amm/MinterAmm.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider addin
g a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:40:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint id;
             ^^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:43:17: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        address proposer;
                ^      ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:46:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint eta;
             ^ ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:49:19: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        address[] targets;
                  ^     ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:52:16: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint[] values;
               ^    ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:55:18: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        string[] signatures;
                 ^        ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:58:17: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        bytes[] calldatas;
                ^       ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:61:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint startBlock;
             ^        ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:64:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint endBlock;
             ^      ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:67:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint forVotes;
             ^      ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:70:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint againstVotes;
             ^          ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:73:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        bool canceled;
             ^      ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:76:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        bool executed;
             ^      ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:79:38: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        mapping (address => Receipt) receipts;
                                     ^      ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:85:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        bool hasVoted;
             ^      ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:88:14: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        bool support;
             ^     ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol:91:16: Warning: Only state variables can have a docstring. This will be disallowed in
 0.7.0.
        uint96 votes;
               ^   ^
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/GovernorAlpha.sol: Warning: SPDX license identifier not provided in source file. Before publishing, con
sider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more infor
mation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/SirenToken.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consid
er adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informat
ion.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/governance/Timelock.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider
 adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informatio
n.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/libraries/Math.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider addi
ng a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/market/IMarket.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider addi
ng a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/market/IMarketsRegistry.sol: Warning: SPDX license identifier not provided in source file. Before publishing, cons
ider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more inform
ation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/market/Market.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider addin
g a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/market/MarketsRegistry.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consi
der adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informa
tion.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/proxy/Proxiable.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider add
ing a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/proxy/Proxy.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding
a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/IERC20.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider addi
ng a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/IRewardsDistribution.sol: Warning: SPDX license identifier not provided in source file. Before publishing,
 consider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more i
nformation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/IStakingRewards.sol: Warning: SPDX license identifier not provided in source file. Before publishing, cons
ider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more inform
ation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/Owned.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider addin
g a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/Pausable.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider ad
ding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/RewardsDistribution.sol: Warning: SPDX license identifier not provided in source file. Before publishing,
consider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more in
formation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/RewardsDistributionRecipient.sol: Warning: SPDX license identifier not provided in source file. Before pub
lishing, consider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org fo
r more information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/SafeDecimalMath.sol: Warning: SPDX license identifier not provided in source file. Before publishing, cons
ider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more inform
ation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/StakingRewards.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consi
der adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informa
tion.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/staking/VestingVault.sol: Warning: SPDX license identifier not provided in source file. Before publishing, conside
r adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informati
on.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/test/MockPriceOracle.sol: Warning: SPDX license identifier not provided in source file. Before publishing, conside
r adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more informati
on.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/test/TestUpgradeableAmm.sol: Warning: SPDX license identifier not provided in source file. Before publishing, cons
ider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more inform
ation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/test/TestUpgradeableMarket.sol: Warning: SPDX license identifier not provided in source file. Before publishing, c
onsider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more inf
ormation.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/test/TestUpgradedMarketsModule.sol: Warning: SPDX license identifier not provided in source file. Before publishin
g, consider adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more
 information.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/token/ISimpleToken.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider
adding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information
.
,/root/workspace/AT 535_SirenMarkets/core 737dad7d3f6b25f9d910a2c851173d7d5c4dd800/contracts/token/SimpleToken.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider a
dding a comment containing  SPDX License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@chainlink/contracts/src/v0.6/interfaces/AggregatorV3Interface.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Identi
fier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/GSN/Context.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Identi
fier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/Initializable.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Iden
tifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/access/AccessControl.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX Licen
se Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/access/Ownable.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Ide
ntifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/math/Math.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Identifi
er: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
```

```
,@openzeppelin/contracts ethereum package/contracts/math/SafeMath.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Iden
tifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/token/ERC20/ERC20.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License
Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/token/ERC20/ERC20Burnable.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX
License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/token/ERC20/ERC20Pausable.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX
License Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/token/ERC20/IERC20.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License
 Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/token/ERC20/SafeERC20.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX Lice
nse Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/utils/Address.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Iden
tifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/utils/EnumerableSet.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX Licens
e Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/utils/Pausable.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX License Ide
ntifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.
,@openzeppelin/contracts ethereum package/contracts/utils/ReentrancyGuard.sol: Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing  SPDX Lice
nse Identifier: <SPDX License>  to each source file. Use  SPDX License Identifier: UNLICENSED  for non open source code. Please see https://spdx.org for more information.

> Artifacts written to /tmp/test 20201111 273 1vlvt4k.6bau
> Compiled successfully using:
     solc: 0.6.12+commit.27d51765.Emscripten.clang

completed initial migration
Deploying SimpleToken logic contract
completed SimpleToken logic contract deploy
Deploying Market logic contract
completed Market logic contract deploy
Deploying MinterAmm logic contract
deploying price oracle
completed price oracle deploy
completed MinterAmm logic contract deploy
Deploying MarketsRegistry logic contract
completed MarketsRegistry logic contract deploy


  Contract: Deposit Limits
     ✓ Enforces Limits (1184ms)

  Contract: AMM Verification
     ✓ Initializes (162ms)
     ✓ Provides capital without trading (545ms)
     ✓ Provides and immediately withdraws capital without trading (349ms)
     ✓ Provides capital with trading (879ms)
     ✓ Buys and sells bTokens (869ms)
     ✓ Withdraws large share of LP tokens (632ms)
     ✓ Sells more bTokens than wTokens in the pool (778ms)
     ✓ Enforces minimum trade size (55ms)
     ✓ Works in initial state (105ms)

  Contract: Minter AMM Expired
     ✓ Expired OTM with constant price (1065ms)
     ✓ Expired ITM with exercise (1627ms)
     ✓ should claimExpiredTokens succeed (745ms)
     ✓ claimAllExpiredTokens should succeed (734ms)

  Contract: Minter AMM Gas Measurement
**************************************************
Measuring gas for AMM with 1 markets:
**************************************************
Initial LP deposit:                                93,362
bTokenBuy 1:                                       275,035
bTokenBuy 2:                                       200,035
bTokenSell 1:                                      202,529
bTokenSell 2:                                      187,529
Post sale LP withdrawal:                           199,228
Post sale LP deposit:                              142,886
Post expiry withdrawal 1:                          151,878
Post expiry withdrawal 2:                          92,139
     ✓ Measures gas for single market (1239ms)
**************************************************
Measuring gas for AMM with 2 markets:
**************************************************
Initial LP deposit:                                93,362
bTokenBuy 1:                                       277,088
bTokenBuy 2:                                       202,088
bTokenSell 1:                                      204,581
bTokenSell 2:                                      189,581
bTokenBuy 1:                                       277,100
bTokenBuy 2:                                       202,100
bTokenSell 1:                                      204,593
bTokenSell 2:                                      189,593
Post sale LP withdrawal:                           297,822
Post sale LP deposit:                              236,318
Post expiry withdrawal 1:                          210,068
Post expiry withdrawal 2:                          135,590
     ✓ Measures gas for 2 markets (2319ms)
**************************************************
Measuring gas for AMM with 4 markets:
**************************************************
Initial LP deposit:                                93,362
bTokenBuy 1:                                       279,141
bTokenBuy 2:                                       204,141
bTokenSell 1:                                      206,634
bTokenSell 2:                                      191,634
bTokenBuy 1:                                       279,153
bTokenBuy 2:                                       204,153
bTokenSell 1:                                      206,646
bTokenSell 2:                                      191,646
bTokenBuy 1:                                       279,153
bTokenBuy 2:                                       204,153
bTokenSell 1:                                      206,646
bTokenSell 2:                                      191,646
bTokenBuy 1:                                       279,153
bTokenBuy 2:                                       204,153
bTokenSell 1:                                      206,646
bTokenSell 2:                                      191,646
Post sale LP withdrawal:                           435,006
Post sale LP deposit:                              329,797
Post expiry withdrawal 1:                          291,060
Post expiry withdrawal 2:                          186,605
     ✓ Measures gas for 4 markets (3896ms)
**************************************************
Measuring gas for AMM with 8 markets:
**************************************************
Initial LP deposit:                                93,362
bTokenBuy 1:                                       283,247
bTokenBuy 2:                                       208,247
bTokenSell 1:                                      210,740
bTokenSell 2:                                      195,740
bTokenBuy 1:                                       283,259
bTokenBuy 2:                                       208,259
bTokenSell 1:                                      210,752
bTokenSell 2:                                      195,752
bTokenBuy 1:                                       283,259
bTokenBuy 2:                                       208,259
bTokenSell 1:                                      210,752
bTokenSell 2:                                      195,752
bTokenBuy 1:                                       283,259
bTokenBuy 2:                                       208,259
bTokenSell 1:                                      210,752
bTokenSell 2:                                      195,752
bTokenBuy 1:                                       283,259
bTokenBuy 2:                                       208,259
bTokenSell 1:                                      210,752
bTokenSell 2:                                      195,752
bTokenBuy 1:                                       283,259
bTokenBuy 2:                                       208,150
bTokenSell 1:                                      210,643
bTokenSell 2:                                      195,643
bTokenBuy 1:                                       283,259
bTokenBuy 2:                                       208,150
bTokenSell 1:                                      210,752
bTokenSell 2:                                      195,752
bTokenBuy 1:                                       283,150
bTokenBuy 2:                                       208,150
bTokenSell 1:                                      210,643
bTokenSell 2:                                      195,643
Post sale LP withdrawal:                           709,734
Post sale LP deposit:                              516,764
Post expiry withdrawal 1:                          565,425
Post expiry withdrawal 2:                          396,400
     ✓ Measures gas for 8 markets (7140ms)

  Contract: AMM Upgradeability
     ✓ Fail to upgrade prior to initialization (47ms)
     ✓ Fail to upgrade from non owner account
     ✓ should upgrade and be able to call function on upgraded contract (124ms)

  Contract: AMM Verification: Oracle
     ✓ should calculate correctly when USDC is the collateral token (1013ms)
     ✓ should calculate correctly when WBTC is the collateral token (1022ms)

  Contract: Volatility Factor
```

```
    ✓ Enforces Limits (158ms)

  Contract: Governance Verification
    ✓ Initializes governance and creates a market (632ms)

  Contract: Market Verification
    ✓ Can close out a market (608ms)

  Contract: Market Fees
    ✓ Sends fees to the owner account (773ms)

  Contract: Market Scenarios
    ✓ Calculates call option decimals for wBTC and USDC at $20k strike (607ms)
    ✓ Calculates put option decimals for USDC and wBTC at $12k strike (660ms)

  Contract: Proxy Market Verification
    ✓ Initializes (330ms)
    ✓ Calculates open and expired state (126ms)
    ✓ Mints option wTokens and bTokens (380ms)
    ✓ Blocks redeem if expired (273ms)
    ✓ Blocks redeem if prior to start of the exercise window (287ms)
    ✓ Allows redeem for European style option (582ms)
    ✓ Allows redeem for American style option (628ms)
    ✓ Allows claiming after expiration with no redemptions (540ms)
    ✓ Allows claiming after expiration with full redemptions (552ms)
    ✓ Allows claiming after expiration with partial redemptions (573ms)
    ✓ Allows closing a position while open (472ms)
    ✓ Sets restricted minter (460ms)

  Contract: Create Markets
    ✓ Creates (273ms)

  Contract: Destroy Markets
    ✓ Destroys (299ms)
    ✓ Sweeps payment or collateral token (270ms)

  Contract: MarketsRegistry Verification
    ✓ Initializes (47ms)
    ✓ Allows owner to update MarketRegistry contract fields (390ms)
    ✓ Allows owner to update the market module itself (185ms)
    ✓ Allows token recovery (214ms)
    ✓ Calculates assetPair correctly for markets and amm (483ms)

  Contract: Market Upgradeability
    ✓ Fail to upgrade from non owner account
    ✓ should upgrade and be able to call function on upgraded contract (47ms)

  Contract: Staking Verification
    no vesting
      ✓ should distribute SI token correctly with all stakers joining prior to reward distribution (655ms)
      ✓ should distribute SI token correctly with 1 staker joining after reward distribution (623ms)

  Contract: Proxy Token Verification
    ✓ Initializes (84ms)
    ✓ Mints (113ms)
    ✓ Transfers (113ms)
    ✓ Burns (103ms)
    ✓ Admin Burns (97ms)
    ✓ Minting and burning should be gated to role (114ms)
    ✓ Destroys (129ms)


  60 passing (1m)
```

Code Coverage

The coverage report could not be obtained because the solc compiler failed to compile the contract files modified by solidity-coverage. Error messages shown as below.

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

```
a4d3d474f67fd8981c801ab4f1c38c4ebb834d5e459bd3c1000a4aa65a46d634 ./contracts/Migrations.sol
0b8c71c9be48d79768ba5077c18ea997eb982dcbaf484e83e3fcbbe2b3ae76ef ./contracts/token/ISimpleToken.sol
80862f0d666a92780520618b0f3693d003a9554039f46888299fc49b46eac32e ./contracts/token/SimpleToken.sol
bb5be64ddcedfe11460c328200932f44a02f61852c360a1d9247ef6e525650e9 ./contracts/test/MockPriceOracle.sol
b5d728b8c37e73624fdebb72e3b697be100a77c8f0c9b89d94928de7f5a1188c ./contracts/test/TestUpgradeableAmm.sol
6b16becf38e1ce750a7a67488df8136d9c96d39dcf82d88123d7610b91c4caca ./contracts/test/TestUpgradeableMarket.sol
380c9cd31a2f6ce28df3bac2564fa724944ab0c0d2351791bda40745a740921c ./contracts/test/TestUpgradedMarketsModule.sol
f6ba66cf8271a8400e17e4877497ec3887ff6d894d7b43bdb9a28cb0e62918c8 ./contracts/staking/IERC20.sol
8c1397e1109259bb434a7cd4e3cdd210ec3f5dd51c08acf41db1e7a54234f5b1 ./contracts/staking/IRewardsDistribution.sol
68ce594f1bd1a73b90d08a060e819cd60229cb9679e0f80ae5c50478ba069bfc ./contracts/staking/IStakingRewards.sol
c7403811c8e54775623a382ac191bec833e062a7418763a34c13500a569b46ef ./contracts/staking/Owned.sol
7704092837e8248a0b4a3c8b54704388085d38be33a6108aa33068cf36ce1640 ./contracts/staking/Pausable.sol
a11510d9060cd2f9bc18fcbc4a6096435a07795ff36a48cca7b0e244770164d8 ./contracts/staking/RewardsDistribution.sol
ecb3bdaf6f5dfa0cbcd209857272ae65a28573efb49ed730ec476fd8905adab0 ./contracts/staking/RewardsDistributionRecipient.sol
b1d05c27b81c3dc5ee4d01da01e6d22d72b525da12c13938c790916e488b7cc8 ./contracts/staking/SafeDecimalMath.sol
f256803411918d5382a87924c38b1b417e0823a46125eddd07911973aa0f61f4 ./contracts/staking/StakingRewards.sol
a9e0eecc78f096fcbd2be0cb319a7b716f75e0f9fb93328bbfbb587789b7e284 ./contracts/staking/VestingVault.sol
51f3be5de20c749efb61d70649ffbc6679d086660d3d9295794648bd708e1ce5 ./contracts/proxy/Proxiable.sol
ed2b5ee22433ccb2e0d00aff2b761e98718a5868b312eab0988a3ca06d04dd2e ./contracts/proxy/Proxy.sol
5e09e6ba636b0afb1155da45c2d5e09067d4f2e3b3b1a9c205ebb3541a08f93d ./contracts/market/IMarket.sol
49e3c63f274360a16fac2a72e75b963f01afd069d320e07682beb1dffcd5f5b2 ./contracts/market/IMarketsRegistry.sol
d3e2fec507079b63e9b05fc70ce7e161d1ddbc1336312a956be59ba722423e7e ./contracts/market/Market.sol
5086745c328c895ba34ed3c5cd514f8babf3e7db10f034a4bc55ef095362cb2d ./contracts/market/MarketsRegistry.sol
b373f1afd9c59fd4ffd56f20f7cce70b30ac2266a3238df2c8781ab9f3096c83 ./contracts/libraries/Math.sol
628bd8af5084c8a9cf7b7776d53b4f9f682c83eabbd4a6549072e451e79f0422 ./contracts/governance/GovernorAlpha.sol
f4eef1bbc60bfbcdb86f6e5025168b80ab83cea991ee6fc9e3835620d347c127 ./contracts/governance/SirenToken.sol
f3913c29e51e755c2fd23f4eb8772fcd32e8117c7b3078dc33d2e223c3d5d0d3 ./contracts/governance/Timelock.sol
f3f52c2da6d0b1371ee7754fbe3ac54e691665c927ade3e961d1e00a9176c9f3 ./contracts/amm/InitializeableAmm.sol
7d7a45948cf0cdfadc1e6fbc6920b28181303116787df83397d16fa73691aa7d ./contracts/amm/ISirenTradeAMM.sol
db47999ca0ea6c131326496bdce6fd45e194572c249b0b53fb2151df2a26ac8b ./contracts/amm/MinterAmm.sol
```

Tests

```
312b11139f177868d3339caadad3d6aa9f6b2171895fb24692fa34fe5ea57822 ./test/testHelpers.js
3a32bfb34be0d383b11b9db794aa629f878dd72c9ba3311772eb08958d2ccd55 ./test/util.js
48dabfb3ca68ef43c4155c0c7205eb79ce82d5389177f81618608b0289fb31f9 ./test/token/proxyToken.js
e3ee2ea73843b0d9270caf4684672b260edd695aea8dd360a0fb931a12acc8df ./test/staking/sirenStaking.js
16e40ef213ae0abe2a15ef50f4375c9eeb388b46e89567ab4032f4d875a68953 ./test/marketsRegistry/createMarket.js
73572d73e04b4a1289f0ccb70b79a609e977c609ce39e0409b3a2a5a37ac82fa ./test/marketsRegistry/destroyMarket.js
1f1aeb7b3d01b840c33dddaa8a677931895bf9773d6f244f6b997b55a1acacb6 ./test/marketsRegistry/proxyMarketsRegistry.js
2a0771886524fedd73a62bf537bdb4c9ddedb8175d5d2f18333ba35bdfc754ed ./test/marketsRegistry/upgradeableMarket.js
bc2697a08d29e35112642bebb566f7470e295f6fd7f85994c24ebdc45dc6e10f ./test/market/closeMarket.js
0240358be2952422b971dc2d9f2c154ffd6590e97928b755b4d8475e6850c5df ./test/market/marketFees.js
68b7a5fc80a4262bdc8cbb2a68e4a55192489b6b583b2e7cb742f92039b88b22 ./test/market/marketScenarios.js
d5227f4e1d7ef435fd2644b8393835a9517fc59b41b9b2b3f2e713175cf795a9 ./test/market/proxyMarket.js
ac7ceffdf9a0cfde175b46efac0730f62ad5bcb88d365b384ead7fcb25dc3756 ./test/governance/governanceScenario.js
1fc1c9c597fd2f48114e514a3efaabc01a6bbb0bd4a565b0cbf85787eb4c6fb8 ./test/amm/depositLimits.js
5b225e04c0000fd7ef2ec0a666f53d45186d8b4961fd69f052f3242aac760b76 ./test/amm/minterAmm.js
c5c1b5637900920ec7b7fae81db14ffc6ba9e8fe7e609cf916728aad01f6560e ./test/amm/minterAmmExpired.js
e649af6f33f4bc9666e93fce37d64c8da422c630374c0ce99473f80e2f7de06b ./test/amm/minterAmmGas.js
fdd79911a3f2646311760a05540d2e7a25368fd185750a2bb276442233b69423 ./test/amm/minterAmmUpgradeable.js
aa1f1a9afbb704b2ea8a13955b37ca5252c19105723167d84c8eeb7d676a36c3 ./test/amm/priceOracle.js
3238c15d3bcd1a640275962a3b16c0e0ed45171fbf338fe5625b25fbf4555337 ./test/amm/volatilityFactorUpdate.js
```

Changelog

- 2020-11-06 - Initial report
- 2020-11-17 - Reaudit report
- 2020-12-11 - Reaudit 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.