Lyra Newport Audit (Summary)

Lyra, 02 December 2022

Confidential
17 Tyrwhitt Avenue, Johannesburg, 2196, South Africa
hello@iosiro.com

Summary

iosiro was commissioned by Lyra to perform a smart contract audit of their latest additions to their options trading platform. The assessment was performed from the 16th of November to the 2nd of December 2022, consuming a total of 20 resource days. The audit uncovered 2 high risk issues, and 3 medium risk issues, along with a number of low and informational risk issues. The Lyra team addressed all the findings prior to the conclusion of the audit.

Scope

The primary focus of the audit was Lyra's Newport update, which most notably expands the Option Markets to support assets other than those issued by Synthetix. This allows Lyra to expand to networks where Synthetix is not available. This was done in preparation for deploying the protocol on Arbitrum and to make use of GMX as a drop-in replacement for Synthetix.

Other notable changes included in the release were:

1. Introduction of a long scale factor, reducing the amount payed out when settling long positions under specific market conditions.
2. Overhaul of core contracts to support tokens with differing decimal precision.
3. Abstracting the exchanging and hedging functionality to support both Synthetix and GMX as secondary markets.

The initial audit scope included the functionality to hedge using GMX's perpetuals exchange. However, this component was under active development throughout the project and only best-effort recommendations for GMXFuturesPoolHedger.sol were provided.

Details

Project Name: lyra-finance
Initial Commit: 90fcb51
Final Commit: 522830d
Files: `BaseExchangeAdapter.sol, GMXAdapter.sol, GMXFuturesPoolHedger.sol, LiquidityPool.sol, LiquidityToken.sol, OptionGreekCache.sol, OptionMarket.sol, OptionMarketPricer.sol, OptionToken.sol, ShortCollateral.sol, BlackScholes.sol, ConvertDecimals.sol, FixedPointMathLib.sol, GWAV.sol, Math.sol, PoolHedger.sol, SimpleInitializeable.sol`

Findings

#	Status	Risk Rating	Title	Reference	Description	Recommendation	Fixed
5.1.1	CLOSED	HIGH	Incorrect interface	IVault.sol#L120 IVault.sol#L121	Prototypes for `getFundingFee` and `getPositionFee` of IVault do not match the Vault implementation on Arbitrum.	Update `IVault` interfaces to match the implementation on Arbitrum.	4f2b4ab
5.1.2	CLOSED	HIGH	Rounding of user funds	OptionMarket.sol#L1190 OptionMarket.sol#L992	Decimal conversion rounds down to 0 when amounts are small enough. As a result, sufficiently small deposits will pull zero tokens from the user.	`_transferFromQuote` and `pendingCollateralConverted` should round up instead of down to ensure a user cannot increase their position, without transferring any additional tokens.	522830d

#	Status	Risk Rating	Title	Reference	Description	Recommendation	Fixed
5.2.1	CLOSED	MEDIUM	Stale Chainlink Price	GMXAdapter.sol #L174	No staleness check is performed on the price received from ChainLink aggregators in `_getChainlinkPrice()`. Prices for assets could therefore be out of date when a board expires or when a user opens or closes a position.	Check whether a price update occurred within the heartbeat interval of the queried price feed. Critical functions such as `settleExpiredBoard`, `openPosition`, `closePosition` and `liquidatePosition` should revert if the price is stale. The heartbeat interval of each price feed is documented on the Chainlink website.	The Lyra team acknowledged the risk, but explained that the protocol prioritizes allowing users to exit their positions if the Chainlink network fails to post updates in a timely manner.

#	Status	Risk Rating	Title	Reference	Description	Recommendation	Fixed
5.2.2	CLOSED	MEDIUM	Unable to withdraw minimum deposit	LiquidityPool.sol #L308	`LiquidityPool.initiateWithdraw()` compares the exact liquidity token amount the user wants to withdraw to the `minDepositWithdraw` setting, whereas `LiquidityPool.initiateDeposit()` compares the USD amount the user wants to deposit against minDepositWithdraw. As a result, unless the liquidity token value is less than 1 USD, a user depositing the minimum amount will not be able to withdraw their liquidity.	`amountLiquidityToken` should be converted to its USD equivalent prior to comparing it to `minDepositWithdraw`.	f22fa99
5.2.3	CLOSED	MEDIUM	ProtectedQuote not always updated	LiquidityPool.sol #L436	When processing the withdrawal queue a partial withdrawal or invalid withdrawal will return and bypass the recalculation of `protectedQuote`.	`processWithdrawalQueue` should break instead of return inside the for loop to ensure `protectedQuote` is updated.	3073cad

#	Status	Risk Rating	Title	Reference	Description	Recommendation	Fixed
5.3.1	CLOSED	LOW	Incorrect amount for approval	LiquidityPool.sol #L795	`LiquidityPool.reclaimInsolventBase()` uses freeLiquidity without converting to the correct decimals when calling `quoteAsset.approve()`	`LiquidityPool.reclaimInsolventBase()` should convert freeLiquidity using `ConvertDecimals.convertFrom18()` prior to calling `quoteAsset.approve()`.	3073cad
5.3.2	CLOSED	LOW	Integer Division	GMXAdapter.sol #L239, GMXAdapter.sol #L263	Loss of precision when dividing and then multiplying.	Rework statements to always first multiply then divide.	4f2b4a
5.4.1	CLOSED	INFO	Inconsistent error handling	GMXFuturesPoolHedger.sol#L928 GMXFuturesPoolHedger.sol#L948	`_hasPendingIncrease` and `_hasPendingDecrease` still make use of `require` statements with string messages.	`GMXFuturesPoolHedger.sol` should use a consistent error format. Custom errors should be defined and used, instead of `require` statements.	4f2b4ab
5.4.2	CLOSED	INFO	Abstract contract	BaseExchangeAdapter.sol#L20	`BaseExchangeAdapter` is intended to be a base contract and should not be deployable.	Add the abstract keyword to `BaseExchangeAdapter`	4f2b4ab

#	Status	Risk Rating	Title	Reference	Description	Recommendation	Fixed
5.4.3	CLOSED	INFO	Make consistent use of IERC20Decimal inteface	GMXFuturesPoolHedger.sol#L85 GMXFuturesPoolHedger.sol#L87	`GMXFuturesPoolHedger` makes use of `ERC20` contract for `quoteAsset` and `baseAsset`, whereas the rest of the code base makes use of `IERC20Decimal`.	Declare `baseAsset` and quoteAsset as type `IERC20Decimal`.	4f2b4ab
5.4.4	CLOSED	INFO	Documentation incorrectly assumes USD as Quote Asset		Throughout the codebase the documentation as prices being in terms of USD, which might not be the case for all future markets. The Newport release enables markets to be any combination of assets, given suitable Chainlink price feeds are available and the secondary market supports exchanging and hedging with the chosen assets.	Update all references to USD in the core contracts to reference quote asset instead.	4f2b4ab

#	Status	Risk Rating	Title	Reference	Description	Recommendation	Fixed
5.5.5	CLOSED	INFO	Index useful event parameters	BaseExchangeAdapter.sol#L237	The BaseExchangerAdapter.MarketPausedSet event does not index the `contractAddress` parameter, even though it should only contain discrete values.	Add the indexed keyword to the `contractAddress` parameter.	4f2b4ab
5.5.6	CLOSED	INFO	Code duplication		The function _abs() is reimplemented multiple times in GMXAdapter.sol, GMXFuturesPoolHedger.sol, OptionMarketPricer.sol, ShortPoolHedger.sol and SNXFuturesPoolHedger.sol. Same is true for `_min()` and `_max()`.	To avoid code duplication, reused functions should be implemented as internal pure functions in a Library.	4f2b4ab
5.5.7	CLOSED	INFO	Code readability		Ternary expressions are regularly used in `LiquidityPool` to constrain a value to an upper limit. The `_min()` function implements the same logic and generally results in better code readability.	Use `_min()` instead of ternary expression to constrain values in `LiquidityPool`.	4f2b4ab

#	Status	Risk Rating	Title	Reference	Description	Recommendation	Fixed
5.5.8	CLOSED	INFO	Unused variable	GMXAdapter.sol #L292	`GMXAdapter.exchangeToExactBaseWithLimit()` never sets quoteSpent but rather returns `quoteNeeded`. This causes `quoteSwapped` in `QuoteSwappedForBase` to always be zero.	Rename the return value of `GMXAdapter.exchangeToExactBaseWithLimit()` to `quoteNeeded` and remove the additional `quoteSpent` variable.	522830d
5.5.9	CLOSED	GAS	Unnecessary cross contract call	GMXAdapter.sol #L157 GMXAdapter.sol #L166	The `GMXAdapter` contract regularly performs a cross contract call to retrieve `PRICE_PRECISION`, which is defined as a constant in the `GMX_Vault`.sol contract.	Store `PRICE_PRECISION` as constant in `GMXAdapter`.	ee94c49
5.5.10	CLOSED	GAS	Define and call internal function inside of notPause modifier.	LiquidityPool.sol #L707, LiquidityPool.sol #L719, LiquidityPool.sol #L973, LiquidityPool.sol #L980, LiquidityPool.sol #L1064	When adding a modifier to a function, the logic within the modifier is inlined. If the modifier is used more than once, this results in repeated code at the bytecode level.	Define an internal `BaseExchanger._notPaused()` function that is called by the `notPaused` modifier to prevent increasing the bytecode size of the contract.	4f2b4ab