Lyra V2

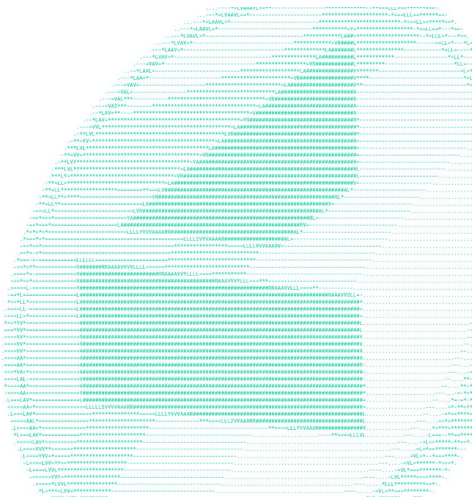
Sean Dawson Dominic Romanowski Anton Cheng Vladislav Abramov

with

Joshua Kim Jake Fitzgerald Nick Forster Timothy Gorham Ksett

July 2023

LYRA V2
WHITEPAPER



Abstract

In this paper, we introduce the Lyra Protocol, a generalized risk engine running on Lyra Chain - an Ethereum rollup built with the OP Stack. A fundamental redesign of previous versions, the Lyra Protocol offers a capital efficient, modular and rapidly upgradable platform for traders, all whilst ensuring that users maintain self-custody of their funds.

1 Introduction

Since launching in 2021, Lyra V1 [1] and its subsequent updates Avalon and Newport have been the leading decentralized options exchange with over $1.5 billion of notional volume traded.

Though an incredible feat, comparing these numbers to those in the traditional world is humbling. Volumes at such exchanges often range into the tens, if not hundreds, of billions per day. Lyra's volumes must therefore grow by several orders of magnitude in order to compete with those in the centralized space.

In this paper, we detail the Lyra Protocol, otherwise known as V2. This is a fundamental redesign of Lyra, one which we believe will be the bedrock of a fair, democratized and decentralized financial system that can compete with traditional entities.

The Lyra protocol can be summarized with the following key features:

- *Generalized Risk Engine*: Any financial instrument (option, dated futures, binaries, etc) can be traded, margined, settled and liquidated on the Lyra Protocol. At launch, support will be provided for European options (with capital efficient spreads), perpetual futures and spot assets.
- *Capital Efficiency*: Traders will have access to sophisticated margin methodologies at launch,

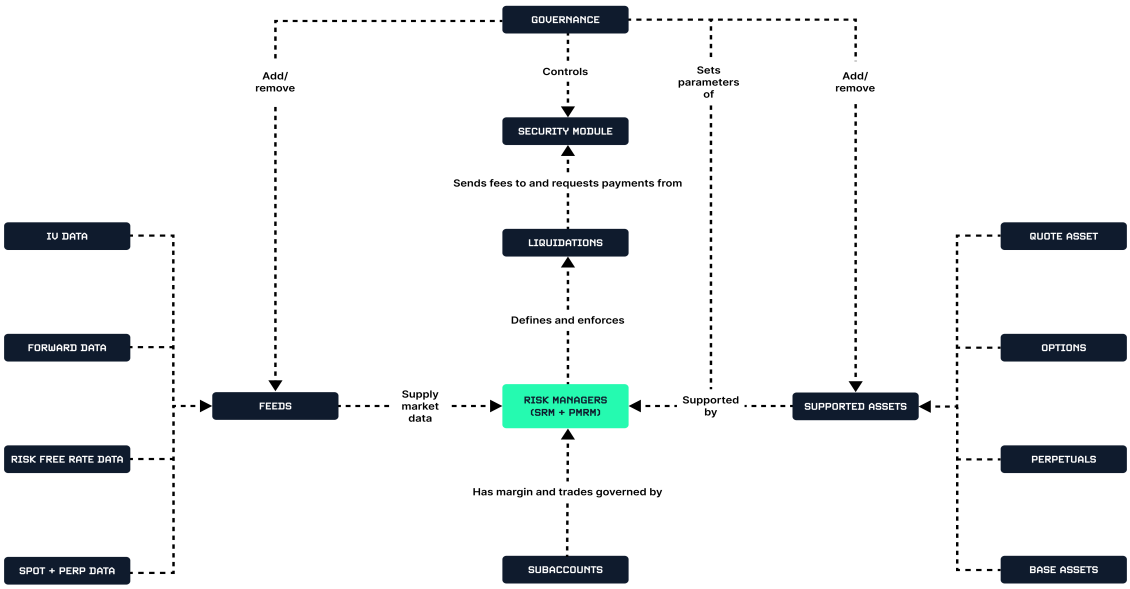


Figure 1: High level schematic of the Lyra Protocol.

unlocking substantial improvements in capital efficiency compared to previous versions.

- *Modularity*: Lyra's smart contracts define various key features like account structure, risk management and a wide array of supported assets. These building blocks can be replaced or built upon more easily than previous versions of Lyra.

2 The Protocol

The Lyra Protocol is a collection of smart contracts on Lyra Chain, an Ethereum rollup on the OP Stack governed by the DAO. There are three main components to the Protocol: subaccounts, managers and assets. The Protocol is modular by design, allowing for swift iteration, meaning many of these layers will be further enhanced over time. For a visual representation of the protocol, see Figure 1.

3 Subaccounts

Subaccounts are the base unit of the Lyra Protocol. All users will be able to permissionlessly mint a subaccount entity which houses all of their supported assets. In turn, all of a subaccount's assets will be governed by a manager, explained below.

Subaccounts are ERC-721 compliant NFTs which store a list of {`asset`, `subId`, `balance`} where:

- the `asset` is a smart contract which encodes rules that define the basic properties of an instrument (such as the fact that an option requires a strike price). Assets are extensible and can be programmed for any purpose. E.g. dated futures and binary option assets can be added at a later date.

- `subId` is a unique label that specifies the parameters necessary to fully define the instrument. E.g. the `subId` of an option will contain information about its strike, expiry and whether it is a call or put.

- `balance` is the net balance of the instrument. This can be either positive (a credit) or negative (a debit)

The subaccounts contract has two main functionalities:

1. Ensure that only authorized parties (the user, managers) are able to change the subaccount's balance.

2. Inform all relevant parties about any trade involving the subaccount.

This was made possible through the use of `hooks`; external function calls to relevant contracts that occur during and after any asset transfer. They are:

- `asset hook`: this invokes the asset to validate the transfer and determine the final balance. For example, an asset contract may apply interest accrued to a transfer event, as well as monitor the time since the last such payment.
- `manager hook`: this calls the manager to validate the final state of the subaccount. The managers can enforce different rules by reverting transfers that make the final state invalid. The manager of a subaccount might read all necessary data from oracles or asset contracts to calculate margin requirements.

This architecture allows different managers and assets the flexibility to create unique validation rules for various subaccount actions.

4 Assets

An asset contract is responsible for determining the results of a transfer, trade, deposit, withdrawal or settlement. To support deposits and withdrawals, as well as interest accrual, an asset has the privilege to update its own balances in any subaccount.

Assets and managers are linked; certain assets will only be supported by specific managers and vice versa. This also means that assets can be fungible across managers. This relationship is maintained via the aforementioned `hooks`.

At launch, there will be four types of assets available:

- `cashAsset`: Each manager supports one `cashAsset` (say, USDC). This is often referred to as the *quote asset*.
- `option`: European options on a given underlying asset.
- `perp`: Perpetual futures on a given underlying asset. Unlike other exchanges, perpetuals can be settled by the user at any point in time. This is referred to as *continuous settlement*. Further, funding paid or received by the user is also continuously applied.
- `wrappedERC20`: Wrapped ERC20 assets such as wBTC. Balances for wrappedERC20s can only be credits.

Assets can also assist the managers at settlement or during subaccount state validation. For example, the `CashAsset` contract stores all relevant data about a subaccount's accrued interest. The manager may want to consider this interest when assessing the risk of a portfolio.

4.1 CashAsset

Each risk manager supports a single quote asset. As with options and perpetuals, users can hold credit or debt balances of the quote asset. Unlike these other assets, users with debit balances will have to pay interest on this sum. This interest will then be distributed to all other users with credit balances. Thus, the quote asset facilitates a native lending market over its supported managers.

At launch, USDC will be the only supported quote asset.

For example, if ETH has a spot price of $2000, then a user holding one ETH in their account can borrow, say, $1600 of USDC using their ETH as collateral. In doing so, they enter a debit balance and pay 5% interest. All other users with credit USDC balances will receive a share of this interest payment.

The interest rate is set via a simple piece-wise curve like in Aave (see [3]). Quote utilization will be defined as the ratio of total debit balances over total credit balances. When utilization is low, the interest rate paid by borrowers will be low. As utilization increases, the so too will the interest rate. Past a certain threshold, the rate of change in interest accelerates greatly.

5 Managers

Managers are smart contracts that specify certain rules for subscribed subaccounts. At launch, there will be two managers available, which ensure that subaccounts maintain appropriate margin when transacting. These are the standard risk manager (SRM) and the portfolio margin risk manager (PMRM). When creating a subaccount, the user must subscribe to a manager. We stress that managers are modular. One major benefit of this design choice is that risk managers can be easily designed for any set of assets or purpose, allowing for diverse trading and effective risk management.

Managers are given the privilege to alter a subaccount's balances since they are responsible for both trade settlement and liquidations. Since managers are smart contracts and are therefore immutable, this privilege cannot be used maliciously.

In the Lyra Protocol, risk managers fulfil the following functions:

1. *Setting Margin Rules*: Managers set the margin requirements of all supported subaccounts. To avoid being subject to liquidation, users must satisfy

their *maintenance margin* mandate. To open a new position, the user must fulfil their (stricter) *initial margin* requirements. These rules are transparent and verifiable by anyone.

2. *Liquidations*: Managers define the conditions for and enforce liquidations.
3. *Settlement*: Managers are responsible for settling all trades.

Both the SRM and PMRM will have (adjustable) open interest caps on supported base assets, options and perps.

5.1 Standard Risk Manager

The standard risk manager gives traders an intuitive, feature rich entry point for trading and margining options, perpetual futures and their base assets on the Lyra protocol. The SRM supports the following instruments:

- Quote asset (USDC at launch)
- Various base (ERC20) assets (such as wETH, wBTC)
- European options and perpetual futures on these base assets

Key features of the SRM include

1. *Cross-margin*: traders use all quote and base balances as collateral for all open positions in a subaccount. Isolated margin is obtained by opening a separate subaccount.
2. *Multi-asset collateral*: traders will be able to use various base assets to collateralize their perpetuals and options. For instance, a user will be able to collateralize a short BTC perpetual with ETH.
3. *Capital Efficient Spreads*: Normally, the margin required for short options is the mark-to-market value of the option and a percentage of the spot. Long options provide no margin offset while base assets are valued at a discount to their market value. To facilitate capital efficient spreads for traders, the SRM is able to cap the margin requirement of spreads at their maximum loss. For example, consider Alice's portfolio consisting of a long $200 call spread, i.e. 1 long ETH $1800 call expiring in 2 weeks (normally requiring $0 of margin) as well as 1 short ETH $2000 call with the same expiry (typically requiring $320 of margin). The SRM will require precisely $0 of margin - a huge improvement in capital efficiency over previous versions. For a short spread, the maximum margin required will be capped at the maximum loss.
4. *Leveraged Perpetuals*: Perpetuals can be traded using the SRM with high capital efficiency.
5. *Borrowing*: If enabled, users can borrow the quote asset against their wrapped ERC20 base assets.

If relevant data feeds have *low confidence*, then the SRM will require extra initial margin for all base assets, perpetuals and short options. In addition, if the quote asset depegs, then further margin will be required for all perpetuals and short options.

5.2 Portfolio Margin Risk Manager

The PMRM brings portfolio margin, an industry standard for allowing high levels of capital efficiency, to Lyra.

The PMRM will support the quote asset, the (single) denominated base asset, as well as corresponding options and perpetuals. Consequently, if a user wishes to use portfolio margin to trade ETH options and BTC options, then two PMRM subaccounts will be required.

To compute the portfolio margin, the manager finds `maxLoss`, the market loss the portfolio would suffer under various scenarios, where each scenario is specified by a given spot and implied volatility shock. Extra *contingency margin* is added to account for risk factors not captured in `maxLoss`, i.e.

$$\texttt{Margin} = \texttt{mfactor} \times (\texttt{contingencies} + \texttt{maxLoss})$$

where `mfactor` is typically 1.0 when computing maintenance margin and a larger value for initial margin. The manager uses the following contingencies:

- *Base Asset Contingency*: A small percentage of the spot price per base asset held.
- *Perpetual Asset Contingency*: A small percentage of the spot price for each perpetual contract held.
- *Option Contingency*: For each strike, the net number of short options is computed. A small percentage of the spot is added to the margin requirement for each net short option.
- *Oracle Contingency*: A large percentage of the spot price is added to the initial margin requirements per base, option and/or perpetual if the relevant data feeds have a low confidence score.

There is also a fifth contingency; the *basis contingency* which computes the loss the portfolio will suffer if the forward basis moves against the user for all expiries. If greater than `maxLoss`, then the basis contingency replaces `maxLoss` in formula for margin.

Finally, if the quote asset depegs, then `mfactor` will increase but only when computing the initial margin.

More detail on the methodologies for the SRM and PMRM will be provided in upcoming documentation.

5.3 Marking and Settlement

Both managers mark options using the `Black76` pricing model, see [2] for more detail.

To ensure traders can smoothly unwind their hedges, options on both managers settle to a time weighted average price (TWAP) of the underlying price in the 30 minutes leading up to expiration.

In this period, options will be marked to a linear combination of the forward and TWAP prices with weights determined by proximity to expiration. Since part of the settlement price will be fixed, this means options will experience so-called *delta decay*. I.e. a normally 100 delta call 15 minutes from settlement will only have 50 delta of underlying exposure.

5.4 Risk Reducing Trades

For both managers, a subaccount cannot open new positions or increase the size of existing positions when the final state has insufficient initial margin. That said, if a transaction results in a reduction in the subaccount's maintenance margin requirements (i.e. the transaction reduces risk), then the update is always permitted.

5.5 Risk Assessors

Due to the large number of `Black76` function calls, portfolio margin computations are very gas intensive. To avoid limiting PMRM account sizes, entities approved by governance called *trusted risk assessors* will be able to bypass all but one risk (namely, the mark-to-market, i.e. solvency) check on chain. While it is possible for trusted risk assessors to approve transactions that result in liquidatable (but not insolvent!) positions, such accounts can be immediately liquidated by any willing participant, removing this risk from the system. Further, the trusted risk assessor role can be revoked if the party in question is found to be acting improperly.

It is important to stress that, due to the solvency check, trusted risk assessors can never allow insolvent positions to be opened.

6 Partial Liquidations

Users who fall beneath their margin requirements will be subject to a partial liquidation.

Liquidations will be a transparent process and open to all participants, i.e. any user can act as a liquidator and the user being liquidated will be able to monitor their subaccount throughout this process.

The liquidation mechanism is designed to expeditiously remove risk from the system while at the same time ensure the most beneficial user experience for traders being liquidated.

6.1 Liquidation Auctions

The way liquidations will work for both managers is as follows.

1. A user is flagged as being under their maintenance margin requirements. Any user can flag this subaccount by calling the function `startAuction()`.
2. The entire portfolio is put up for auction. The initial offer is a small percentage discount to its mark-to-market value.
3. This discount increases over time. At any point, a user (liquidator) can agree to take on any percentage of the portfolio at the current discounted mark-to-market value.
4. To ensure liquidated users get a fair deal, the maximum percentage a liquidator can take is a function of the current discount, portfolio mark-to-market value and current margin requirements. I.e. a portfolio just shy of its margin requirements will only need, say, 10% liquidated while an account near insolvency will require nearly 100% to be liquidated.
5. The discount increases until a sufficient amount of the portfolio is liquidated (by potentially many liquidators) or the discount reaches 100% (where a liquidator can take on up to all of the portfolio for free).
6. An insolvent auction then begins, where bidding starts at \$0 and decreases up to a large (negative) value based on the subaccount's initial mark-to-market value. Liquidators can always take up to 100% of the portfolio and get paid by the Security Module (see below) based on the current (negative) offer.

6.2 The Security Module

The Security Module (SM) is a smart contract that will hold reserves of the quote asset (i.e. USDC) to pay out funds to liquidators in the event of an insolvency.

Should the SM run out of funds to do so, the insolvent debt will have to be shared amongst all users.

This will be done via a *temporary withdrawal fee*. The insolvent debt of the system, as a fraction of sum of the insolvent debt and total deposited (i.e. wrapped) quote asset, is applied as a fee to users wishing to withdraw their quote asset.

For example, say there is \$10,000 of bad debt and \$100,000 of USDC in the system. A fee of

$$\frac{10,000}{100,000 + 10,000} = 9.09\%$$

is applied to all withdrawals. So, if Alice wants to withdraw \$5000, then she pays a \$454.5 withdrawal fee.

The temporary withdrawal fee provides no incentive for being the first to withdraw funds and also prevents liquidation cascades, thereby increasing the system's integrity.

7 Protocol Fees

Every risk bearing entity in the architecture can receive fees for doing so. The protocol has three main sources of fee generation:

1. *Trading fees*: All transactions that increase the open interest (and therefore risk) of the system will be charged a fee. This fee will be a function of the spot price of the underlying asset and size of the trade.
2. *Liquidation fees*: Users who are liquidated will be charged a small fee that scales with the risk and market value of their subaccount.
3. *Interest rate spreads*: A percentage of all interest paid by quote asset borrowers is taken as a fee.

All fees will accrue to the Protocol owned Security Module which improves the robustness and resiliency of the protocol.

Entities approved by governance can bypass the trading fee in exchange for, say, a different fee scheme pledge. These approvals can be rescinded if the entity does not meet its requirements.

8 Parameters and Governance

There are many parameters that govern the protocol; this is especially true for the managers. These parameters also have to be updated whenever there is a market shift to ensure both capital efficiency and security for traders.

All parameters in the protocol are controlled by a smart contract. In Lyra's case, the smart contract will be owned by the short term executor. For details on the short term executor, see [4].

9 Data Feeds

The managers, in order to accurately assess and control risk, require oracles to provide the following data:

1. *Implied Volatility (IV)*: IV for supported tenors will be provided as raw stochastic volatility inspired (SVI) surfaces. These are specified by 5 parameters: (a, b, ρ, m, e). See [5] for more detail. Consider a strike K with log-moneyness k, where $k = \log(K/F)$ and F is the current forward price with time to expiration τ. The implied volatility σ of strike K is given by

$$\sigma = \frac{1}{\sqrt{\tau}}\sqrt{a + b[\rho(k-m) + \sqrt{(k-m)^2 + e^2}]}.$$

2. *Forward*: The forward price for supported tenors is also supplied. As mentioned, close to expiration, this becomes a linear combination of the TWAP of the underlying and forward prices.
3. *Risk Free Rates*: This is an interest rate curve defined at each supported tenor.
4. *Spot*: This is the spot (index) price of the underlying assets sourced from multiple outlets.
5. *Perpetual Price*: This is the trading price of the perpetual future.

Each piece of data supplied also has an associated *confidence score*. Low confidence feeds result in extra initial margin being required for both the SRM and PMRM.

More detail on the methodology for deriving the implied volatility and forward bases will be announced shortly.

References

[1] *Lyra Whitepaper. 2021, S. Dawson, N. Forster, D. Romanowski, M. Spain, https://www.lyra.finance/files/whitepaper.pdf.*

[2] *The pricing of commodity contracts* 1976, Journal of Financial Economics, 3, 167-179, F. Black

[3] *Borrow Interest Rate* Aave V3 documentation https://docs.aave.com/risk/liquidity-risk/borrow-interest-rate

[4] *LEAP 51: Lyra Governance V2 2*023, M. Spain, https://leaps.lyra.finance/leaps/leap-51/

[5] *Arbitrage-free SVI volatility surfaces J.* Gatheral, A. Jacquier, Quantitative Finance, Vol. 14, No. 1, 59-71, 2014