Drift Protocol v0

Drift Labs

Working Copy - August 2021

Abstract

Drift Protocol is a decentralized exchange for perpetual futures built on Solana. It features cross-margined trading accounts and a virtual AMM (vAMM) for price discovery. This paper offers a brief technical overview of the v0 architecture before providing a more in-depth description of the protocol's core features.

Contents

1 Technical Overview

The following section provides a technical overview of the Drift Protocol v0. Later sections dive deeper into specific features such as the vAMM and cross-margining. **Figure 1** displays the v0 on-chain architecture.

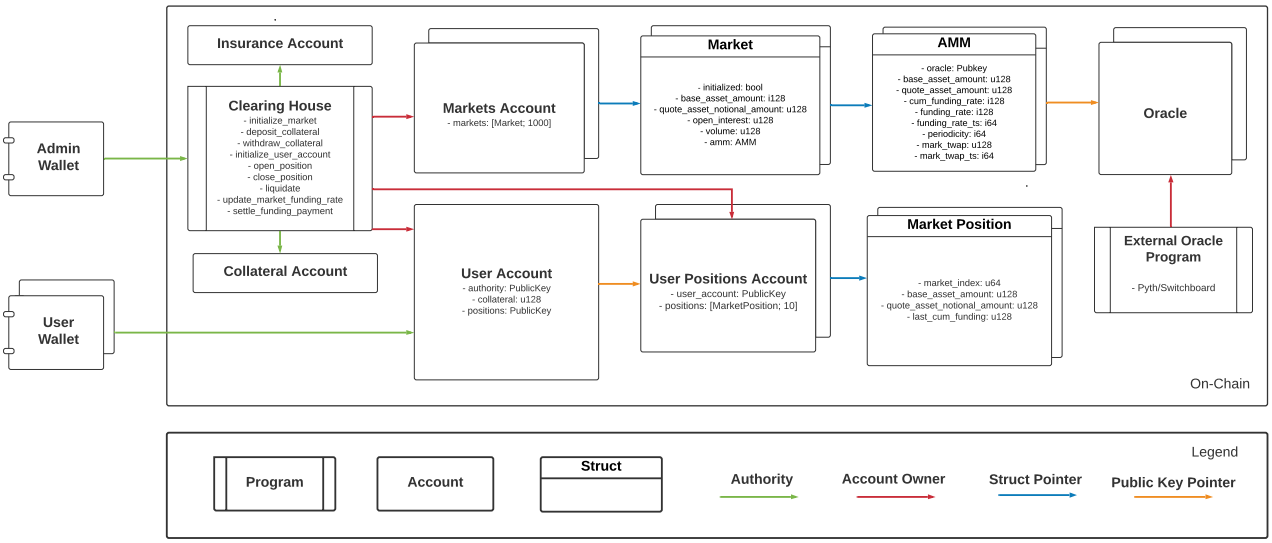

Figure 1: Drift Protocol v0 On-Chain Architecture

1.1 Components

Drift Protocol contains the following core components:

- **Clearing House**: Facilitates user interactions with the exchange
- **Global Collateral Vault**: Escrows all of the user collateral deposits
- **User Account**: Records the state of a user's account with the Clearing House (e.g. collateral deposited, realized PnL)
- **User Market Position**: Records the state of a user's position in a market (e.g. the size of their position)
- **Markets**: Record the state of a market (e.g. open interest). Contains a reference to vAMM and Oracle
- **Virtual AMMs (vAMM)**: Provides a price discovery mechanism for Perpetual Future mark prices
- **Oracles**: Provides the price used to calculate Perpetual Future funding payments
- **Insurance Fund**: Pays for user withdrawals when the Global Collateral Vault has insufficient funds. Financed via trading and liquidation fees.

1.2 Participants

The exchange is designed with an admin and three types of users in mind:

- **Admin**: Initializes the insurance fund and new markets. Modifies Clearing House and market parameters. To be under control of a DAO.
- **Traders**: Users that are interested in speculating in the future price of a market or running quantitative trading strategies.
- **Arbitrageurs**: Delta-neutral users that are interested in collecting spreads between our exchange and other exchanges, playing an active role in pushing the platform mark price towards oracle prices and collecting funding rate payments
- **Liquidators**: Users that are willing to run liquidations on under-collateralized users in exchange for a reward.

1.3 Instructions

The Clearing House program enables the following user interactions:

1. **User Account Initialization**: Create a User Account with the Clearing House
2. **Market Initialization**: Initializes a new Perpetual Futures market. Admin only.
3. **Deposit Collateral**: Deposit collateral in the Global Collateral Vault
4. **Withdraw Collateral**: Withdraw an amount of collateral less than or equal to the user's free collateral
5. **Open Position**: Open a long or short position in a Perpetual Future. The position size is bounded by their free collateral
6. **Close Position**: Close out a user's Perpetual Future position and attributes the realized PnL to the user's collateral
7. **Liquidate**: Close out a user's positions, transferring a portion of the remaining collateral to liquidator and portion to the liquidator. User must be below maintenance margin ratio
8. **Update Market Funding Rate**: Update a market's funding rate. Success is conditional on a sufficient amount of time elapsing since the previous funding rate update
9. **Settle Funding Payment**: Settle a user's outstanding funding payments. If the user is owed funding payments, the insurance fund pays the users; if the user owes funding payments, the user pays the insurance fund

1.4 Off-Chain Dependencies

Drift Protocol is dependent on two actions being triggered off-chain by exchange participants:

1. **Market Funding Rate Updates**: Market funding rates can be updated once an hour by any participant. Traders and Arbitragers are incentivized to update the market rate so that they receive their funding rate payments
2. **Liquidations**: Liquidations can be triggered by any user. Users are incentivized to liquidate overleveraged positions as the liquidator receives a fraction of the liquidated users' collateral after the position is closed

1.5 Data Structures

```
pub struct UserAccount {
    pub authority: Pubkey,
    pub collateral: u128,
    pub positions: Pubkey,
}

pub struct UserPositionsAccount {
    pub user_account: Pubkey,
    pub positions: [MarketPosition; 10],
}

pub struct MarketPosition {
    pub market_index: u64,
    pub base_asset_amount: i128,
    pub quote_asset_notional_amount: u128,
    pub last_cum_funding: i128,
}

pub struct MarketsAccount {
    pub markets: [Market; 1000],
}

pub struct Market {
    pub initialized: bool,
    pub base_asset_amount: i128,
    pub quote_asset_notional_amount: u128,
    pub open_interest: u128,
    pub volume: u128,
```

```
    pub amm: AMM,
}

pub struct AMM {
    pub oracle: Pubkey,
    pub base_asset_amount: u128,
    pub quote_asset_amount: u128,
    pub base_asset_price: u128,
    pub peg_multiplier: u128,
    pub cum_funding_rate: i128,
    pub funding_rate: i128,
    pub funding_rate_ts: i64,
    pub periodicity: i64,
    pub mark_twap: u128,
    pub mark_twap_ts: i64,
}
```

2 Virtual Automated Market Maker (vAMM)

2.1 Market Definition

Every market is implemented as a perpetual futures contract, priced using a virtual AMM.

2.1.1 Constant Product Curve

The current implementation is a two-dimensional constant product curve:

$$x \cdot y = k$$

The market price for x in terms of y is the slope of the curve, given the relative scarcity:

$$price = \frac{y}{x} \cdot 1$$

Using Drift's naming conventions, we have:

$$\underbrace{base\_asset\_amount}_{\text{underlying asset}} \cdot \underbrace{quote\_asset\_amount}_{\text{USDC}} = k$$

Given that the market is virtual, there is no liquidity pool and there are no tokens for users to hold. Instead, users' positions are stored in a Market Position struct owned by the Clearing House. When a user places an order, their respective base or quote asset amounts are swapped with the vAMM's. During the swap, the pair updates to maintain the constant product invariant and the price moves accordingly. The size of the user's position is constrained by their margin ratio (discussed in **Section 4**).

These markets are imbued with value from the Clearing House's insurance fund which pays out imbalance in funding rates (**Section 5**) and losses incurred from leveraged activity. The insurance fund is initialized by the admin and is further funded by liquidation fees. The liquidation mechanism and fee structure are discussed in **Section 6**.

2.1.2 Initializing k and *peg_multiplier*

At the time of initialization, t_{init}, we initialize the balances as follows in order to concentrate liquidity at the oracle price:

$$base\_asset\_amount = quote\_asset\_amount =: \sqrt{k}$$

Since the ideal k is not deterministic at t_{init}, we set a single market's k in expectation of Participant demand in that market. To protect the insurance fund from paying excessive $funding\_rate$ imbalance, the protocol errs to initialize with conservatively low k, with plans to increase k as evidence for consistent Participant demand is revealed.

In order to peg this to a starting oracle price, we introduce a *peg_multiplier* to pin onto a *price* value. For maximum liquidity / minimum slippage, the multiplier is initialized around an oracle twap price at t_{init}.

$$price = \frac{quote\_asset\_amount}{base\_asset\_amount} \cdot peg\_multiplier$$

The Clearing House tracks profit and losses(PnL) using this price formula above. The effective price a user enters a position with, is the average price along the curve. This means, before explicit protocol fees, a user opening then closing a single position, with no protocol events between, will not see any PnL.

2.1.3 Adapting k and *peg_multiplier*

The admin account has the sole authority to modify these parameters as needed. The impact on the Clearing House is a function of the market's bias. We define bias as the long-short delta, i.e. the cumulative user position:

$$\begin{aligned} bias &= market.base\_asset\_amount \\ &= \sum_i user\_position_i.base\_asset\_amount \end{aligned}$$

In the event that bias = 0, we can freely adjust k and *peg_multiplier* without any dislocations in cumulative user PnL (given path independence). Since an individual user PnL can be altered, it would be done only to support healthier market conditions.

2.2 Future Implementation

2.2.1 Alternative pricing curve

Drift Protocol plans to test the benefits of other pricing curve constructions that offer more flexibility and concentration liquidity around the peg price.

2.2.2 Formulaic k and *peg_multiplier*

In future versions we aim to optimally adjust these parameters in additional scenarios, where bias is non-zero.

For instance, since every order is a taker order, volume can be broken up into arbitraging and non-arbitraging volume. Arbitraging volume is defined as volume that moves the AMM's price towards the oracle price. A high appetite for arbitraging volume per oracle price movement encourages us to support a higher k, providing lower slippage for Traders.

3 Trading and Order Specifications

3.1 Current Implementation

3.1.1 Market Orders

User trades are executed against a vAMM. Drift Protocol supports market orders along with the ability to specify a limit price. The limit price provides a mechanism for users to specify the maximum slippage that

they're willing to pay. If the order is executed and the price is worse than the limit price, the transaction is canceled.

Although one can calculate the vAMM's slippage at any given time, one can't guarantee the order of the transactions so a user can receive a price worse than expected when they initiate a transaction.

All orders are either traded within a single transaction or they are canceled. No orders are queued, so if the transaction isn't filled instantly, it will be canceled and users will have to replace their orders.

3.2 Future Implementation

3.2.1 Limit Orders

Future versions of Drift Protocol may support a matching engine for limit orders. One potential solution involves adding an order book to the exchange and incentivizing off-chain bots to match orders against the vAMM when prices cross the limit order price. These will resemble limit orders, stop losses and take profits in practice.

4 Margin Methodology

The Margin Method defines the rules for Traders to take take additional leverage using margin managed by the Clearing House.

4.1 Current Implementation

Drift Protocol v0 supports a rules-based cross-margining multiple positions using a single pool of USDC collateral. Figure 2, describes some trade offs between **Cross Margin**: a combined margin ratio across all open positions and **Isolated Margin**: which separates margin for each positions into separate collateral account.

Margin Method	**Pros**	**Cons**
Isolated Margin*	Position blocks can limited loss to capital pool.	Excess collateral may not be swept pre-liquidation.
Cross Margin	Profits from one position can support losses in another.	All collateral is at risk of liquidation.

Figure 2: Margin Methods Comparison
(*: Not available in v0)

4.1.1 Definitions

With multiple positions in the cross-margin implemention, the margin ratio can be calculated as:

$$margin\_ratio = \frac{total\_collateral}{position\_notional}$$

where:

$$\begin{aligned}
total_collateral &= user_account.collateral + unrealized_pnl \\
unrealized_pnl &= \sum_i position_value_i - user_position_i.quote_asset_notional_amount \\
position_notional &= \sum_i position_value_i \\
position_value_i &= user_position_i.base_asset_amount \cdot amm.base_asset_price
\end{aligned}$$

4.1.2 On-Chain Limitations

The two limiting factors for implementing cross-margining on Solana were the maximum transaction size and the instruction compute unit limit. Solana transactions must be smaller than 1232 bytes. An account address is 32 bytes, thus the maximum number of accounts that can be included in a transaction is 40. If each Market were stored in its own account, a user would be constrained to entering 40 markets. This is because when a user interacts with the exchange (e.g. opens a position, withdraws collateral), the protocol must check the user's margin ratio which requires fetching the price of each market.

Instead of storing the state of each market in one account, Drift Protocol stores the state of all its markets in a single account. This is possible because vAMMs have a space-complexity of O(1) (whereas an order book has a space-complexity of O(n) where n is the number of orders).

$$\begin{bmatrix}
index & base_asset_amount & ... \\
i_0 & ... & ... \\
i_1 & ... & ... \\
i_2 & ... & ... \\
... & ... & ... \\
i_{998} & ... & ... \\
i_{999} & ... & ...
\end{bmatrix}
\quad
\begin{bmatrix}
index & market_index & ... \\
n_0 & i_0 & ... \\
n_1 & i_1 & ... \\
n_2 & i_2 & ... \\
... & ... & ... \\
n_8 & i_8 & ... \\
n_9 & i_9 & ...
\end{bmatrix}$$

Figure 3: MarketsAccount (supports 1000 markets), UserPositionsAccount (supports 10 user positions)

The instruction compute unit limit is the more constraining factor. As mentioned above, every time a user interacts with the exchange, the exchange must calculate the user's margin ratio. This requires looping through each of the user's positions and thus, given enough positions, can consume all of an instruction's compute units. To start, Drift Protocol enables a user to enter 10 positions. This a conservative start and

more testing is required to determine an upper bound. Solana's increases from a 200,000 compute unit limit to 1,000,000 compute unit limit will also increase the number of positions a user can enter.

4.2 Future Implementation

4.2.1 Isolated Margin

To enable support for isolated margin accounts, the protocol would enable users to create multiple UserPositionsAccounts (these accounts could also be named UserPositionGroups) and specify what amount of their total collateral should be allocated to each positions account.

4.2.2 Risk-based Margin Method

To bring about additional capital efficiency, the protocol intends to add a risk based margin method. Features include:

- Allowing different margin requirements across different markets;
- Reducing collateral required for holding correlated long/short positions;

4.2.3 Indefinite Number of Markets

To extend to an indefinite number of markets, the protocol can be modified to support an arbitrary number of MarketsAccounts.

5 Funding Rates

Funding Rate payments are the incentive mechanism to bring the vAMM's mark price closer to the oracle price. For instance, a user with long position in a market whose $oracle\_price > mark\_price$ will receive a payoff proportional to their position size.

5.1 Current Implementation

For each market, every hour, the Funding Rate update transactions are initiated by users who are incentivized to collect the funding rate payment. This update reflects the premium or discount between the vAMM's mark price and the oracle's price on average over the previous hour window.

5.1.1 Definitions

To enable funding rate payments lazily, Drift Protocol stores a cumulative funding rate since initialization.

$$amm.cum\_funding\_rate_0 = 0$$

$$...$$

$$amm.cum\_funding\_rate_t = \sum_t \frac{amm.mark\_twap_t - amm.oracle.twap_t}{amm.oracle.twap_t \cdot 24}$$

A user position account then stores the cumulative funding rate they have received so far in each market ($position_i.last\_cum\_funding$). All funding rate payments are settled directly with the user's collateral. The Clearing House automatically attempts to settle the user's funding rate payments upon user interaction (withdraw, open_position, close_position, etc). Updating the funding payment owed is as follows:

$$P_{i_{ba}} = position_i.base\_asset\_amount$$

$$P_{i_{fr}} = position_i.last\_cum\_funding$$

$$M_{i_{fr}} = amm_i.cum\_funding\_rate$$

$$user.collateral \mathrel{+}= \sum_i P_{i_{ba}} \cdot (M_{i_{fr}} - P_{i_{fr}})$$

After the funding payment is completed, the user's last cumulative funding rate is updated to match the market's:

$$P_{i_{fr}} = M_{i_{fr}} \forall_i$$

In the event that collateral deposits do not cover withdrawals of funding rate payments, the Insurance Fund covers the remainder.

5.2 Future Implementation

5.2.1 Continuous Funding

Future versions of Drift Protocol can explore a more continuous approach to funding rate updates. Instead of updating the funding rate once an hour, funding rates may be updated every time a user interacts with the vAMM, i.e. every time they open or close a position. In addition to making the funding rate more precise, this would remove the off-chain dependency where users must trigger funding rate updates once an hour.

5.2.2 Asymmetric Funding

Currently, funding rates are symmetric amongst longs and shorts. Mechanisms around the funding rate imbalance may also be altered to protect the insurance fund.

6 Liquidations

6.1 Current Implementation

Liquidations are a necessary function for the risk management of margin trading. The margin rules are defined in **Figure 4**:

Requirement	**Margin Ratio <**	**Implication**
initial_margin_requirement	.20	No risk increasing *open_positions*
partial_margin_requirement	.0625	Vulnerable to 25% position reduction 2.5% remaining collateral penalty
maintenance_margin_requirement	.05	Vulnerable to 100% position reduction 100% remaining collateral penalty

Figure 4: Drift Protocol v0 Margin Requirements/Liquidation Rules

Anyone can run a liquidator bot to reduce/close positions of a user whose *margin_ratio* is below the *partial_margin_requirement*. The remaining collateral penalty is split equally between the insurance fund and liquidator. Leveraged losses beyond 0% margin are solely absorbed by Drift's Insurance Fund.

We include partial liquidations to reduce the likelihood of users getting entirely liquidated by offloading positions earlier. A user in multiple positions will have each position reduced by 25% of their *position_notional* size.

A user is fully liquidated at a 5% maintenance margin. At this point, a liquidator closes out 100% of the user's position at the market price and receives the user's remaining collateral.

6.2 Future Implementation

Future work will focus on improving the speed of liquidations and requiring liquidators to put up capital to bolster the Insurance Fund. Examples include:

- Experimentation of dynamic parameters for liquidations and risk management;
- Allowing liquidators to put up a position's initial margin in order to inherit or take over positions instead of directly closing out positions on the market, which reduces pressure on Drift's Insurance Fund;
- New set of liquidation rules under future risk-based margining system

7 Disclaimer

This paper is for general information purposes only. It does not constitute investment advice or a recommendation or solicitation to buy or sell any investment and should not be used in the evaluation of the merits of making any investment decision. It should not be relied upon for accounting, legal or tax advice or investment recommendations. This paper reflects current opinions of the authors and is not made on behalf of Drift Labs or their affiliates and does not necessarily reflect the opinions of Drift Labs, their affiliates or individuals associated with them. The opinions reflected herein are subject to change without being updated.