dYdX V4 Oracle Driven Permissionless Market Listings

Executive Summary

Listing thousands of permissionless assets introduces significant liquidity risk to dYdX. We identified pain points in illiquid markets and proposed the appropriate mitigants.

Pain Point	Mitigant
Increased Price Volatility	Early Deleveraging, Variable Margin Requirements
Ease of Price Manipulation	Dual Price Mechanism

First, illiquid markets have thin and shallow order books, causing significant order slippage. This further increases price volatility and leads to costly liquidations for the insurance fund. To prevent this, we suggest implementing early deleveraging of defaulted contracts. If a contract's estimated liquidation price is worse than its mark price, it is deleveraged instead. This socializes

counterparty risk and reduces the net cost to the insurance fund. Early Deleveraging also reduces an attacker's access to the insurance fund and winds down their position in the process.

Second, illiquid markets are easier to manipulate. Attackers need less funds to move the index price of an illiquid spot market. Instead of relying solely on spot price oracles, we suggest using the Dual Price Mechanism. This mechanism balances the spot price and derivative market price of an asset. With both markets utilized, an attacker's cost of manipulation is increased. They have to significantly manipulate the spot price of an asset or manipulate both the spot and derivative prices.

Third, we set the initial margin requirement of each asset to its 99% VaR. This reduces the number of defaulted positions with negative account balances while giving traders access to leverage.

With these mitigants in place, dYdX can mitigate risks while giving users access to thousands of markets. The mitigants require little capital from dYdX and, thus, are scalable. Mitigants and market listing can be implemented programmatically by utilizing data aggregators like Coingecko and CoinMarketCap. These aggregates provide market data and historical pricing for thousands of assets. The protocol can capture volatility in these markets while reducing counter-party risk.

Introduction

dYdX takes on counter-party risk for all traders. Thus, its primary concern is leveraged positions having negative account value. Leveraged positions can default from natural market volatility or due to market manipulation. Typically, an efficient liquidation engine mitigates most of these risks. In illiquid markets, liquidation loses effectiveness as a mitigant due to increased slippage and price volatility. To mitigate these risks, we propose a series of mitigants:

1. Decoupling Mark Price from Index Price
2. Tearing Up defaulted contracts
3. Variable Margin Requirements

These mitigants are designed around the pain points of illiquid markets. They also fit seamlessly with the existing contract specs utilized by dYdX.

Risk Mitigants

Decoupling Mark Price from Index Price

Mark Prices are estimates of the actual value of a contract and are used to handle liquidations. Currently, dYdX uses the Oracle price to handle liquidations. In illiquid markets, price oracles are prone to manipulation. Traders can manipulate the spot market to extract profit from the derivative markets, as seen with COMP a few years ago. With newly listed assets, there is no spot market to reference. Upon listing, the perpetual market on dYdX would initially have greater volume and liquidity, making the spot market a bad choice for the reference price.

If both derivate and spot markets exist, the mark price should account for both. This is typically done using the Dual Price Mechanism. Under the Dual Price Mechanism, the mark price is calculated as follows:

$$MP = median(p_1, p_2, p_l)$$

Where $p_1 = p_{index}(1 + F)$, $p_2 = p_{index} + MA(p_{mid} - p_{index})$, and p_l is the last traded price on the derivative exchange. The moving average will sample one data point per minute for the last hour. The Dual Price Mechanism is used by exchanges such as Bybit, Binance, and Phemex. Kucoin uses p1, and BTSE uses a price impact modifier of the spot price.

The dual Price Mechanism is an effective attack mitigant as it reduces the effectiveness of manipulating just one of an asset's spot or derivative prices. This can be seen by running through a few scenarios. For all scenarios, we will use the following notation and assumptions:

- All prices, including the Mark Price, are functions of time. So $p_1(0)$ is p1 at time 0
- Manipulation occurs at t=1 and significantly increases or decreases the mark or index price
- MA refers to the moving average of its argument. The time period is not relevant to this proof.
- $p_{mid} \approx p_{index}$ for all time periods prior to t=0 and t=1
    - Thus $MA(p_{mid} - p_{index}) \approx 0$ prior to t=1

    - And $MA(p_{mid}(1) - p_{index}(1)) = \frac{p_{mid}(1) - p_{index}(1)}{n}$ where n is the # of points in the MA
- $MP(0) \approx p_{index}(0) \approx p_l(0) \approx p_{mid}(0)$
    - It follows that $F(0) \approx 0.0$ and from that $p_1(0) \approx p_{index}(0)$
- $-0.04 \leq F(t) \leq 0.04$
    - See funding rate documentation
    - F is negligible as the scale of price manipulation far exceeds $\pm$4%

WWTS:

1. If $p_{index}(0) \ll p_{index}(1)$ then $MP(1) \leq p_{index}(1)$
2. If $p_l(0) \ll p_l(1)$ and $p_{mid}(0) \ll p_{mid}(1)$ then $MP(1) \leq p_l(1)$

These two cases demonstrate the impact of upward price manipulation on the MP of an asset. Downward price manipulation follows the same process, though inverted.

To demonstrate the inequality between a Mark Price and the index price, we will compare each component price, p_1, p_2, p_l, to the index price.

Case 1: Index price manipulated upward

Given $p_{index}(0) \ll p_{index}(1)$, $p_l(0) \approx p_1(1)$, and $p_{mid}(0) \approx p_{mid}(1)$

WWTS $MP(1) \leq p_{index}(1)$

$p_1(0) \approx p_{index}(0)$

$p_1(1) = p_{index}(1) * (1 + F(1)) \approx p_{index}(1)$

$p_2(1) = p_{index}(1) + MA(p_{mid}(1) - p_{index}(1))$

We need to show that $MA(p_{mid}(1) - p_{index}(1)) < 0$ to demonstrate that $p_2(1) < p_{index}(1)$.

As $MA(p_{mid}(1) - p_{index}(1)) = \frac{p_{mid}(1) - p_{index}(1)}{n}$ and $p_{mid}(1) \ll p_{index}(1)$,

$MA(p_{mid}(1) - p_{index}(1)) < 0$

Therefore, $p_2(1) < p_{index}(1)$

We already know that:

- $p_l(0) \approx p_l(1)$
- $p_l(0) \approx p_{index}(0)$
- $p_{index}(0) \ll p_{index}(1)$

Thus by transitivity, $p_l(1) \ll p_{index}(1)$

$MP(1) = median(p_1(1), p_2(1), p_l(1))$

As $p_1(1), p_2(1), p_l(1)$ are all less than or equal to $p_{index}(1)$, $MP(1) \leq p_{index}(1)$

Case 2: Perpetual Market Price manipulated upward

Given $p_{index}(0) \approx p_{index}(1)$, $p_l(0) \ll p_1(1)$, $p_{mid}(0) \ll p_{mid}(1)$, and $p_{mid}(1) \approx p_l(1)$.

WWTS $MP(1) \leq p_l(1)$

$p_1(0) \approx p_{index}(0)$

$p_1(1) = p_{index}(1) * (1 + F(1)) \approx p_{index}(1)$

As the index price is roughly unchanged, $p_1(1) \approx p_1(0) \approx p_l(0)$

Thus $p_1(1) \ll p_l(1)$

$p_2(1) = p_{index}(1) + MA(p_{mid}(1) - p_{index}(1))$

We know that $p_{index}(1) \ll p_l(1)$ and $p_{mid}(1) \approx p_l(1)$.

We need to show that $MA(p_{mid}(1) - p_{index}(1)) \leq p_{mid}(1) - p_{index}(1)$.

As $MA(p_{mid}(1) - p_{index}(1)) = \frac{p_{mid}(1) - p_{index}(1)}{n}$ and $n \geq 1$,

$MA(p_{mid}(1) - p_{index}(1)) \leq p_{mid}(1) - p_{index}(1)$

Thus $p_2(1) < p_l(1)$

Naturally, $p_l(1)$ is equal to itself.

$MP(1) = median(p_1(1), p_2(1), p_l(1))$

As $p_1(1), p_2(1), p_l(1)$ are all less than or equal to $p_l(1)$, $MP(1) \leq p_l(1)$.

As shown above, the Dual Price Mechanism keeps the MP of an asset from being tied strictly to a manipulated perpetual or spot index price. However, the assumptions used above will only sometimes hold in an illiquid market. Wide bid-ask spreads can cause p_l to deviate from p_{mid}. Volatility in prices can cause $MA(p_{mid} - p_{index})$ to deviate from 0 as well significantly. The Dual Price Mechanism increases the capital requirement for an attack but doesn't prevent them altogether.

Newly Listed Assets

Assets w/out spot markets do not have index prices. Their mark price should instead be calculated as $0.5 * MA(p_l + p_{mid})$, the average of the MA of p_l and p_{mid}. The moving average helps smooth out fluctuations caused by market volatility. It also prevents an attacker from executing a single large trade to suddenly move the mark price.

Early Deleveraging

With an illiquid perpetual market, liquidations are prone to high slippage. This can cause liquidated positions to incur additional losses, which the insurance fund has to cover. Alternatively, positions in need of liquidation can be offset against existing positions. dYdX already has deleveraging in its contracts. Instead of using this as a last resort, liquidated contracts could be covered by users' open positions. This is similar to partial tear-ups done by Clearing Houses, where a defaulted position is offset against equal and opposite positions.

This mechanism should only be invoked when market liquidity cannot absorb a liquidation. If a position's execution price, p, is worse than its mark price, it should be torn up. For longs, we check if $p < MP$; for shorts, we check if $p > MP$. Early deleveraging is initiated if this condition is met *and* the user's margin cannot cover the liquidation (i.e., the insurance fund would be invoked).

To compensate users for absorbing positions, half the liquidation fee (if any) is given to them. The insurance fund still covers any negative balance during deleveraging.

Example scenarios showing when to invoke Early Deleveraging vs. Liquidations.

Liquidating positions will cover most defaults. In cases where it cannot due to market illiquidity, Early Deleveraging reduces or eliminates costs to the insurance fund.

Variable Margin Requirements

The simplest way to minimize loss for the protocol is to tighten margin requirements. Disabling leverage by default would significantly hurt UX. Stringent margin requirements are needed to secure the protocol while allowing leveraged trading. A simple one-size-fits-all solution won't suffice. Instead, the margin should be based on historical price action. A common standard w/ derivatives trading, see section 3.1, is to reference the 10-day value at risk w/ a 99% confidence level over the last year of data. If an asset has less than 90 days of trading data, its initial and maintenance margin requirements default to 40% and 20%, respectively. This is in line with suggestions provided by FINRA Rule 4210 for future contracts.

As traders can open either long or short positions, both tails of the price distribution must be considered in this calculation. Alternatively, margin requirements for long and shorts can differ.

Differing requirements for short vs. long positions is a risk-mitigant technique utilized by futures exchanges. As prices cannot be negative, a 5% loss in price is more significant than a 5% increase. Thus, different margin requirements for short/long positions are a reasonable solution to this issue.

The VaR gives a rough worst-case scenario loss in a single day of holding a position. The initial margin requirement is set to the VaR to minimize insurance fund costs. The maintenance margin requirement is then half the initial margin requirement. This significantly reduces the number of defaults and potential costs to the insurance fund.

Margin requirements will be updated once every trading day. Markets entering periods of increased volatility will be protected by steeper margin requirements. Stable markets are rewarded with lower margin requirements. To keep in line with existing margin requirements, the initial margin requirement should always be at least 10%. This prevents the poor UX of offering users a 1% margin on a highly volatile asset.

Newly Listed Assets

Assets w/out spot markets will have to use their perpetual market data instead. Assets w/out 90 days of trading data should not have leverage enabled for long positions. There is already a competitive advantage to listing an asset before its launch. Thus, it is safer to disable leverage for long positions and require a steep 80% initial margin requirement for short positions. Some leverage is allowed for short positions to facilitate

Implementation

Implementing these mitigants can be handled using data aggregators like Coingecko and CoinMarketCap. They provide detailed data on thousands of assets, including historical prices and sources of liquidity.

Asset Listing

Using the free Coingecko API, we can fetch a list of all supported assets.

```
def fetch_coin_list() -> Collection[Text]:
    """Get the ids of 3 random assets supported by Coingecko."""
    coin_ids = [
        coin['id'] for coin in
        requests.get('https://api.coingecko.com/api/v3/coins/list',
```

```
                    timeout=60).json()]
    random.shuffle(coin_ids)
    return coin_ids[:3]
```

With our asset IDs, we can collect trading data from the last year and calculate the initial margin requirements.

```
def calculate_initial_margin_requirement(
    coin: Text='bitcoin') -> Collection[float]:
    """Calculate the initial margin requirement of an asset.

    Returns a tuple of requirements for longs and shorts."""
    price_data = pd.DataFrame(requests.get(
        f'https://api.coingecko.com/api/v3/coins/{coin}/'
         'market_chart?vs_currency=usd&days=365&interval=daily', timeout=60
    ).json()['prices'], columns=['date', 'price'])

    # Disable long margin and set short margin to 80% for unlisted assets.
    if price_data.empty:
        return (100, 80)
    # Default to 40% for assets w/ less than 90 days of trading data.
    elif price_data.shape[0] < 90:
        return (40, 40)
    else:
        long_requirement = calculate_var(price_data, 99)
        short_requirement = -1 * calculate_var(price_data, 1)

        # Disable margin if requirement exceeds 90%.
        long_requirement = long_requirement if long_requirement < 90 else 100
        short_requirement = short_requirement if short_requirement < 90 else 100
        return
            (long_requirement, short_requirement)
```

The historical method for calculating VaR can be utilized:

```
def calculate_var(price_data: pd.DataFrame,
                  confidence_level: float=99) -> float:
    """Calculate the VAR of an asset."""
    price_data['returns'] = price_data['price'].pct_change()
    price_data['log_returns'] = np.log(1 + price_data['returns'])

    # Convert to percentage.
    return -100 * np.nanpercentile(price_data['log_returns'],
                                   100 - confidence_level, method='lower')
```

Below are three random assets and their initial margin requirements (IMR) generated using the above code:

Asset	Long IMR	Short IMR
Gempad	10.6%	8.7%
Versasity	12.1%	14.5%
BitStubs	40.0%	40.0%

Note the different requirements for short and long positions. As stated before, a single IMR can be calculated instead for ease of user experience. The exception is unlisted assets whose longs have leverage disabled.

Market Management

The implementation of early deleveraging relies on the mark price calculation, so the two are combined here. Similar to the funding rate, the market's mark price is updated once every minute.

```
def update_market(market: Market) -> None:
    """Update the market state and default positions where necessary."""

    # Fetch the latest index, mid, and last traded prices
    market.fetch_prices()
    market.funding_rate = market.update_funding_rate()

    # Update Mark Price
    market.p1 = market.index_price * (1 + market.funding_rate)
    market.p2 = (market.index_price
                 + market.calc_ma(market.mid_prices - market.index_prices))
    market.mark_price = math.median(market.p1, market.p2, market.last_price)

    for position in market.positions:
        if position.margin <= position.maintence_margin_requirement:
            price = market.estimate_execution_price(position.size)
            requires_insurance_fund = market.estimate_liquidaiton_cost(position)

            # If deleveraging is cheaper & liquidating requires the insurance
            # fund to cover the position, then we deleverage the position.
            if ((position.size > 0 and price < market.market_price) or
                (position.size < 0 and price > market.market_price) and
                requires_insurance_fund):
                market.early_deleverage(position)
```

```
            else:
                market.liquidate(position)
```

Once the mark price is updated, every position is checked. If a position is eligible for liquidation, we check if early deleveraging is required —i.e. is the insurance fund losing money liquidating — and preferred. If so, we deleverage. Otherwise, we liquidate.

Conclusion

Insuring counter-party risk for thousands of illiquid assets is a challenging endeavor. Illiquid markets are prone to manipulation and lead to inefficient liquidations. To mitigate these risks, we suggest three simple and scalable mechanisms.

Increase Capital Requirement of Attack

First, to increase the capital requirement of attacks, the mark price of a contract should be calculated using the Dual Price Mechanism. This ties the mark price to trading activity on both perpetual and spot markets, preventing an attacker from simply manipulating the least liquid of the two. The Dual Price Mechanism also utilizes a moving average, smoothing out price volatility.

Improved Liquidation Efficiency

Second, to improve the efficiency of liquidations, we suggest utilizing Early Deleveraging in cases where liquidation prices would cost the insurance fund money. By deleveraging profitable users, we can reduce overall leverage amounts in times of volatility and cut down on costs to the insurance fund. This also reduces an attacker’s ability to drain the insurance fund by minimizing its usage.

Margin Requirements

Last, we suggest strict margin requirements for markets. By basing the margin on the asset’s 99% VaR, we further reduce the need for the insurance fund to be invoked. They also serve as risk indicators to less experienced traders who want to trade rare assets.

These mitigants do not impose strict trading caps or require any capital from the protocol aside from its insurance fund. Any number of markets can be opened and run under this system. This isn’t a risk-free system, but it directly identifies and tackles the pain points of illiquid markets in a way that aligns w/ the system dYdX currently utilizes.