dYdX: A Standard for Decentralized Derivatives

Antonio Juliano

September 25, 2017

Abstract

We present a set of protocols that allow several types of financial derivatives to be created, issued, and traded for any ERC20 token. All described protocols are trustless, fair, and completely free to use. They are intended to serve as an open standard for derivatives created on underlying ERC20 tokens. Our approach uses off chain order books with on chain settlement to allow creation of efficient markets.

Contents

1 Introduction

The rise of blockchains has enabled anyone to own and transfer assets across an open network without needing to trust any external parties. Unlike existing financial architecture, blockchains are freely and equally available worldwide. This has led to a large and rapidly increasing number of digital assets existing on the blockchain. Many centralized and decentralized platforms designed to facilitate the efficient exchange of these assets already exist, and more are in development. Such platforms allow investors to take long positions in various assets. However, it is currently very difficult or impossible to take more complex financial positions.

dYdX allows creation of entirely new asset classes which derive their value from underlying blockchain based assets. Derivatives allow investors to achieve superior risk management with their portfolios, as well as open up new avenues for speculation. Derivatives also increase market efficiency for the underlying asset by aiding in price discovery and allowing individuals to express more complex opinions on price and volatility. dYdX provides advantages over traditional derivatives by eliminating the need for a regulated central clearing house, providing global and equal access, and allowing users full control of their funds at all times.

The size of the derivatives market on existing financial infrastructure far outstrips the market size of any other type of financial asset. It is roughly estimated to be over $1.2 quadrillion[1], or more than 10 times the total world GDP. We believe that as decentralized platforms mature and start to offer significant advantages over traditional financial systems, an ever increasing number of traditional assets will start to be listed on the blockchain.

dYdX will offer a number of decentralized protocols implementing various types of popular derivatives. These protocols are comprised of Ethereum Smart Contracts and standards. The protocols will be governed by a single decentralized entity.

[1] Investopedia. *How big is the derivatives market?*.
http://www.investopedia.com/ask/answers/052715/how-big-derivatives-market.asp

2 Existing Work

There are few existing decentralized derivatives protocols and none that have any significant usage. Centralized exchanges also fail to offer adequate derivatives on decentralized assets. Consequently, it's very difficult to take short or more complex financial positions of the bulk of today's decentralized assets.

In order to enable a decentralized derivatives protocol to operate, there needs to be a way to trustlessly exchange assets, as well as determine the price at which assets will be exchanged. A decentralized exchange protocol is one that facilitates the trustless exchange of one token for another at prices dictated by the market. dYdX uses the 0x protocol[2] to enable token exchange at rates supplied by users of the protocol.

Several types of decentralized exchanges have been proposed: on-chain order books, automated market makers, state channels, and a hybrid off-chain order book approach. The 0x whitepaper offers an in-depth discussion of the tradeoffs between these models[3]. We chose to base dYdX on the hybrid approach pioneered by 0x, as we believe it allows creation of the most efficient markets. 0x allows market makers to sign and transmit orders on an off-blockchain platform, with the blockchain only used for settlement.

One previous attempt at decentralized derivatives, Velocity[4], proposed using an oracle based approach to feed the exchange rates of asset pairs to a smart contract responsible for operation of options contracts. The contract would then use this price information to create and exercise options. Using such an oracle based approach has several significant drawbacks. The limitations on frequency, latency, and cost of price updates due to the nature of blockchains makes it impossible to create markets as efficient as those built on traditional centralized exchanges. Using an oracle also adds a great deal of centralization to any protocol, as some central parties have full control over setting the price. Worse, if those central parties were also trading on the protocol, they would have a huge economic incentive to manipulate prices in their favor.

dYdX protocols allow trade of derivatives at any price indicated in a valid 0x order. This means there is no need for the contracts to be aware of the market price, as they will accept any order. Takers of a derivative provide orders of their choosing, which are then used to execute the exchange for the derivative. It is in the economic interest of the taker to choose an order with the best price. This best price is dictated by the market, and no orders with better prices will exist.

[2] Will Warren, Amir Bandeali. *0x White Paper*. https://0xproject.com/pdfs/0x_white_paper.pdf

[3] Will Warren, Amir Bandeali. *0x White Paper*. "Existing Work". https://0xproject.com/pdfs/0x_white_paper.pdf

[4] Shayan Eskandari, Jeremy Clark, Vignesh Sundaresan, Moe Adham. *On the feasibility of decentralized derivatives markets*. https://users.encs.concordia.ca/~clark/papers/2017_wtsc.pdf

3 Derivatives

dYdX consists of a number of protocols specifying the operation and execution of different types of derivatives. We plan to prioritize the development of the most popular and widely used types. Below we outline our implementation of protocols for options and short sells. We plan to develop protocols for additional types derivatives in the future.

3.1 Option

3.1.1 Description

In an option, a holder of an asset sells the right to buy or sell that asset at a specified strike price and future date[5]. An option to buy an asset is referred to as a call, and an option to sell an asset is called a put. The seller of the option (called the writer) collects a premium upon sale, but is also bound to buy or sell the asset at the agreed upon price and date if the holder of the option desires. A covered option indicates that the underlying asset is put up as collateral, so it is guaranteed to be able to be collected at a future date. The option can itself be traded on the open market. We describe an implementation of an American covered option, or one which can be exercised at any time before the expiration date.

3.1.2 Use Cases

Options enable numerous trading strategies that can be designed for speculation or risk management.

Options can be used to provide additional leverage in speculation. For example suppose the price of AAPL is $100, and an investor who has $1000 to invest believes it will go up. The investor could buy 10 shares at $100, and if he is correct and the price rises to $110, he could sell and will have made a $100 or 10% profit. Suppose instead that the investor had purchased call options with a $100 strike and $2 premium. The investor could afford 500 of these options with his $1000. If the price again rose to $110, the investor could exercise his options to buy at $100, and then immediately sell at $110 for a $10 profit per option. Since the investor had paid $2 for each of these options, he would have made $8 per option on the trade. This means the investor's profit would have been $8 * 500 = $4,000 or a 400% return. This shows how with the same amount of capital investors can achieve much larger returns using options than by simply holding the asset.

Options can also be used to hedge or reduce risk in an investment. Imagine an investor is long 100 share of AAPL, which is again trading at $100. The investor could purchase a put option with $90 strike for a $2 premium. Such an option would ensure that for only a 2% fee, during the lifetime of the option the investor could not lose more than 10% on his investment.

[5] Investopedia. *Options Basics: What are Options?*. http://www.investopedia.com/university/options/option.asp

Options also enable more advanced trading strategies such as straddles, strangles, collars, and many more. Among other things, such strategies can lock in a price, profit from volatility in any direction, or profit from price stability in an asset.

3.1.3 Overview

The dYdX option protocol uses one Ethereum Smart Contract per type of option. A type refers to a given set of input parameters including the *base token*, *quote token*, strike price, and expiration date. *base token* refers to the asset the option is for and *quote token* refers to the token in which the premium and strike price are denominated[6]. Each option contract is able to issue new options of its type at any time before the option expiration date. The contracts can act as either a put or a call option by simply switching the *base token and quote token* and inversing the strike price.

Writers of the option list offers for a specified lot size and premium on an off blockchain platform. Buyers can buy options from a writer by sending a transaction containing a write offer to the smart contract. After receiving such a transaction, the smart contract transfers the premium in *quote token* to the writer, and the offered amount of *base token* to itself. The buyer is issued options which can be transferred and traded as any other ERC20 token. The smart contract holds on to the *base token* until the option is either exercised or expired.

Any holder of the option can choose to exercise at any time before the expiration date. Upon exercise, the option holder pays $strike\ price * (\#\ options)$ of *quote token* to the smart contract and is sent $\#\ options$ of *base token* from the smart contract. After the option expires, all writers can withdraw *quote token* and *base token* from the smart contract corresponding to $\frac{Options\ Written}{Total\ options\ Written} * (total\ tokens\ held)$.

3.1.4 Implementation

3.1.4.1 Contracts

We use three types of smart contracts to allow the issuance and functionality of options: the *Creator*, *Proxy*, and *CoveredOption* contracts.

The *Creator* is responsible for creating all *CoveredOption* contracts. Anyone can create a new type of *CoveredOption* by providing the the following specifications:

- The address of the ERC20 token the option is for (referred to as *base token*)
- The address of the ERC20 token the strike price and premium are to be paid in (referred to as *quote token*)
- The strike price (broken into two parts to form an exchange rate between *base token and quote token*)

[6] Investopedia. *Base Currency*. http://www.investopedia.com/terms/b/basecurrency.asp

- The expiration date

Creating a new type of *CoveredOption* only opens it up for sale, and does not issue any options. There can exist only one *CoveredOption* for each combination of input parameters.

The *Proxy* is responsible for transferring user tokens between accounts. Users use the ERC20 allowance functionality to authorize the *Proxy* to move their tokens. Each new *CoveredOption* is authorized to use the *Proxy* to transfer user funds when it is created by the *Creator*.

The *CoveredOption* contract represents a specific type of covered option. Each one implements the ERC20 interface to allow shares of the option to be traded and transferred after issuance. This means every option can be publicly traded on an exchange as any other ERC20 token.

3.1.4.2 Issuance

CoveredOption uses the exchange functionality of the 0x Protocol to facilitate issuance of new options. Options can be issued anytime before the expiration date of the option. In order to issue new options, the writer broadcasts a signed message in the 0x message format specifying the following information:

- The address of the writer
- The address of the fee recipient
- The amount of *base token* the writer is offering
- The amount of *quote token* to be paid as a premium to the writer upon purchase
- The expiration time for the sale of this option
- The address of the *CoveredOption* contract for the option they want to write. This address is specified in the taker field of the message, so only the *CoveredOption* contract can take the trade

The writer must have at least as much *base token* as offered, and must set allowance on the *Proxy* contract. Buyers can buy less than the amount of options offered by the writer. In 0x terminology, the writer will be the maker of the trade, and the *CoveredOption* contract will be the taker of the trade. The message can be published in any channel, but is a binding agreement to offer the specified sale. Relayers can then list these option sale offers on an option issuance order book (much the same as relayers in the 0x protocol).

When a buyer wants to purchase an option, they send a transaction to the *CoveredOption* contract that includes the message signed and broadcast by the writer, and the amount of options they wish to buy. Options are issued on a 1:1 ratio with the amount of *base token* deposited by the writer. Once the *CoveredOption* contract receives this transaction it does the following:

1. Validates the expiration date of the option has not yet passed
2. Calls into the *Proxy* to transfer the appropriate amount of *quote token* from the buyer to the *CoveredOption* contract itself. This is the premium that is being paid for the option.

3. Call the *0x Exchange Contract* to exchange the *quote token* which was just taken from the buyer with the appropriate amount of *base token* from the writer. The *0x Exchange Contract* validates the the writer's signature, ensuring this offer is legitimate. The writer is the maker and the *CoveredOption* contract is the taker in this trade. After this, the writer ends up with the *quote token* premium, and the *CoveredOption* contract ends up with the offered amount of *base token*. The *CoveredOption* contract will hold the *base token* until the option is settled.
4. The *CoveredOption* contract records that the writer has deposited the amount of *base token*. This amount is used later in the case the option expires without being exercised.
5. The balance of the buyer is increased by the amount of options purchased. The buyer is now the holder of that amount of the options, and can now freely transfer and trade them as per the ERC20 standard.
6. If the amount of options available to be written was less than the amount desired by the buyer, the excess *quote token* left over after the trade is transferred back to the buyer.

All of the above steps happen atomically (i.e. they all happen, or none of them happen) in a single transaction.

3.1.4.3 Exercise

Before the option expires, any holder of the option can exercise any amount less than or equal to the number of options he owns. This means the holder agrees to pay the strike price (globally specified on the *CoveredOption* during its creation), for every option he wishes to exercise. It is only in the holder's economic interest to exercise his options if the market price for the *base token* is greater than the strike price of the option.

In order to exercise, the owner sends a transaction to the *CoveredOption* contract indicating how many options he wishes to exercise. Assuming the transaction is valid, the *CoveredOption* contract:

1. Calls into the *Proxy* to transfer $strike\ price * \#\ options$ of *quote token* from the sender to the *CoveredOption* contract itself
2. Deducts balance from the owner
3. Sends the owner *base token* on a 1:1 basis with number of options exercised
4. Holds onto the *quote token*. The appropriate portion can later be withdrawn by each writer of the option

3.1.4.4 Withdrawal

After the option expires, any writer of the option can withdraw a proportion of both *base token* and *quote token* held by the *CoveredOption* contract corresponding to:

$$\frac{Options\ Written}{Total\ options\ Written} * (total\ tokens\ held)$$

This is done by sending the *CoveredOption* contract a withdraw transaction, which causes the contract to send the writer their full balance of each token, and sets the writer's written balance to zero.

If an address is both the writer and holder of an equivalent number of options, it may at any time withdraw any amount of *base token* less than or equal to:

min(# *options written*, # *options held*)

Doing so will decrease both the address's balance and number of options written by the amount withdrawn. This is provided as a utility so a writer can always get the *base token* back, even before the option expires, by purchasing the desired number of options.

3.1.5 Extensions

3.1.5.1 Naked Options

A naked option refers to one that is not fully collateralized, meaning the writer must be trusted to supply the *base token* as it is exercised. Naked options are more commonly traded than covered options.

The covered option protocol can be relatively easily extended to support naked options. The same approach as above would be used, except the writers would not immediately put up the *base token* as collateral, but would instead be asked to only supply it on exercise of the option. The writers could then choose to either put up the *base token* or default on their side of the contract agreement.

A reputation system is necessary to enable naked options. Reputation could exist external to the dYdX options protocol, and many different implementations could exist. Since all actions on the options protocol are public, it would not be difficult to penalize default. Buyers of options would then price the writer's reputation into their decision on whether to purchase an option from a specific writer. More reputable writers could then charge higher premiums, while less reputable and more risky writers would most likely charge lower premiums.

3.2 Short Sell

3.2.1 Description

In a short sell, an investor borrows an asset and immediately sells it[7]. The asset must be repaid to the lender, usually along with interest, at a later date. The investor makes money if the price of the asset decreases, since when he repays the lender he can buy the asset for less than what he sold it for. The investor loses money if the price of the asset increases, because he must later pay more for the asset than he borrowed it for. The lender makes money from the interest paid by the buyer.

3.2.2 Use Cases

Short sells are used to enable investors to profit from an asset which decreases in price. Short sells can be used for both speculation and hedging. Investors can use a short sell for speculation when they believe the price of an asset will go down. Short sells can be used to hedge existing positions by shorting a correlated asset. Lending assets for a short sell can provide the lender with additional interest from the loan.

3.2.3 Overview

The dYdX short sell protocol uses one main Ethereum Smart Contract to facilitate decentralized short selling of ERC20 tokens. Lenders can offer loans for short sells by signing a message containing information about the loan such as the amount, tokens involved, and interest fee rate. These loan offers can be transmitted and listed on off-blockchain platforms.

When a short seller wishes to initiate a short he sends a transaction to the dYdX short sell smart contract containing a loan offer, a 0x buy order for the token being shorted, and the amount of the short. Upon receiving this transaction, the smart contract transfers the margin deposit from the short seller to itself, and then uses 0x to sell the loaned token using the specified buy order. The smart contract holds onto the deposit and quote token resulting from the sale of the loaned token for the life of the short.

The short is closed when the short seller sends a transaction to the smart contract containing a 0x sell order offering to sell the amount of token borrowed from the lender for an amount less than $deposit + initial\ sell\ amount - interest\ fees$. Upon receiving this transaction, the contract uses 0x to execute the trade between the order maker and itself. After, the contract sends the original amount of the loaned token and the total interest fee amount of the quote token to the lender. The short seller is sent an amount of the quote token equal to $deposit + profit - interest\ fees$. Note the profit could be negative if the base token rose in price.

The loan for a short sale can also be called in by the lender after a specified lockout time. Once the loan is called in, the short seller has a specified amount of time to close the short sell.

[7] Investopedia. *Short Selling*. http://www.investopedia.com/terms/s/shortselling.asp

3.2.4 Implementation

3.2.4.1 Contracts

For short selling, there are four contracts used: the *ShortSell* contract, the *Proxy* contract, the *Repo* contract, and the *Vault* contract.

The *Proxy* is the same one that was described in the Covered Option section, and is used to transfer user funds.

The *ShortSell* contract offers functionality to enable short selling. It contains all the business logic and public functions. The *ShortSell* contract is designed to be upgradable (see the governance section).

The *Repo* contract holds the state for short sells initiated through the *ShortSell* contract. It exposes a simple interface, and only the *ShortSell* contract is authorized to use it. Similarly, the *Vault* contract holds all the funds locked up in short sells. It also exposes a simple interface which the *ShortSell* contract is authorized to use. Separating the state and token holdings from the logic enables the *ShortSell* contract to be easily upgraded without migrating state.

3.2.4.2 Offering Message

The first ingredient to a short sell is a lender who holds the *base token*, and wants to lend it out for a given deposit and fee rate in *quote token*. The lender prepares and signs a message with the following information:

- The address of the *base token*
- The amount of *base token* offered
- The address of *quote token*
- The minimum deposit amount in *quote token*
- The minimum sell price in *quote token*
- The interest rate (denominated in *quote token* / day)
- The fee recipient (for the original sale, not the interest loan. As in 0x protocol)
- The maker fee (paid by lender)
- The taker fee (paid by the short seller)
- An expiration time for the offer
- A lockout time before the loan can be called in (can be 0 if none desired)
- A call time limit (the amount of time between when the lender calls in the loan and when the loan must be repaid)

This message can then be broadcast off blockchain on exchanges similar to 0x protocol messages. It is a binding agreement to commit to the loan if a short seller desires. It is expected that these offers will be listed on exchanges and will compete on interest rate and terms.

3.2.4.3 Buyer

The second ingredient is a buy order which can be filled as part of the short sell. This buy order comes in the form of a 0x order to buy the *base token* for an amount of *quote token*. The buy order can come from anywhere, and the buyer is in no way involved in the loan or short sell contract. This order can be for any price, and must be selected by the short seller. The only prerequisite is the order must be for at least as much of the *base token* as the short seller is shorting. It is in the short seller's economic interest to select the buy order with the best price.

3.2.4.4 Shorting

Once a short seller has selected a loan offering and buy order, the short seller then sends a transaction to the *ShortSell* smart contract containing:

- the signed loan offering
- the signed buy order
- the amount of *base token* the short seller wishes to sell
- the amount of *quote token* the short seller wishes to put up as a deposit

When the contract receives the transaction the following happens:

1. The signature and inputs on the loan message are verified
2. The amount of deposit is validated to be greater than or equal to the minimum deposit rate requested by the lender
3. The *ShortSell* contract calls into the *Proxy* to transfer the offered deposit in *quote token* from the short seller to the *ShortSell* contract itself
4. The *ShortSell* contract calls into the *Proxy* to transfer the requested amount of the *base token* from the lender to the *ShortSell* contract itself
5. The *ShortSell* contract records that the requested amount of the loan has been used, and saves it in a mapping. This is used to keep track of the amount remaining in the loan offer and protect against replay attacks using the signed loan message[8]
6. If there is a taker fee specified in the supplied order, the *ShortSell* contract calls into the *Proxy* to transfer the appropriate fee for the trade from the short seller to itself
7. The *ShortSell* contract calls into the *0x Exchange Contract* to exchange the appropriate amount of *base token* for the amount of *quote token* the short seller is taking. The buyer is the maker in this trade and the *ShortSell* contract is the taker. The *0x Exchange Contract* will verify the inputs and signature on the supplied buy order
8. The details of the short sell are stored in the contract, mapped by a unique public identifier for the short. This identifier is used by the short seller and/or lender to interact with the short sale at a later date

[8] Will Warren, Amir Bandeali. *0x White Paper*. "Fills & Partial Fills". https://0xproject.com/pdfs/0x_white_paper.pdf

All steps happen atomically. At the end, the *ShortSell* contract ends up with an amount of *quote token* equal to the deposit put up by the short seller plus the *quote token* resulting from the sale of the *base token*. The contract holds onto these funds until the loan is repaid by the short seller.

3.2.4.5 Closing

The short seller can decide to close the short sell by presenting the *ShortSell* contract with a 0x sell order to sell greater than or equal to the amount of borrowed *base token* for an amount of *quote token*. This sell order can be for any price such that there is enough *quote token* left over after the sale to pay the lender his full fee, however it is in the short seller's economic interest to select an order with the lowest price. When the *ShortSell* contract receives a transaction containing such a sell order from the short seller, the following happens:

1. The total interest fee (in *quote token*) owed to the lender at this point in time is calculated
2. The *ShortSell* contract calls into the *Proxy* to transfer the fee required for the sell from the short seller to the *ShortSell* contract itself
3. The *ShortSell* contract calls into the *0x Exchange Contract* to execute the trade of *quote token* for the amount of *base token* in the loan. After the trade, the *ShortSell* contract holds the original loaned amount of *base token* and an amount of *quote token* equal to $deposit + short\ seller\ profit$ (profit could be negative)
4. The amount of *quote token* held by the *ShortSell* contract for this short sell is validated to be greater than or equal to the total interest fee
5. The *ShortSell* contract sends *quote token* equal to the total interest fee to the lender
6. The *ShortSell* contract sends the loaned amount of *base token* to the lender
7. The *ShortSell* contract sends *quote token* equal to $deposit + profit - interest\ fee$ to the short seller
8. The *ShortSell* contract deletes the short sell from its storage

At the end of the short, the short seller ends up with $profit - interest\ fee$ more *quote token* then he started with. The lender makes the amount of interest fee in *quote token*. The *ShortSell* contract itself ends up net neutral as desired.

3.2.4.6 Calling

The other way a short sell can be settled is by the lender calling in the loan from the short seller. This can only happen after the specified lockout time on the original loan. It is done by the lender sending the *ShortSell* contract a transaction indicating they are calling in the loan. After this transaction the short seller has the amount of time originally specified in the loan (call time limit) to pay it back. The short seller uses the same process described above in the closing section to close the short. If the short seller fails to close the short the lender is entitled to the entire *quote token* balance locked in the short sell.

It is in the lender's interest to call in the short when the price of *base token* relative to *quote token* rises to the point that the *quote token* locked in the short is almost not enough to buy back the original *base token*. This means the lender needs to be watching the price and be ready to call in the short on an upward price movement. As a utility, the lender can also authorize another address to call in the loan on his behalf. This authorized party would most likely be an exchange or service that watched the price and was always ready to programmatically call in loans on price movements.

This approach also requires that the short seller is always online and able to send a transaction to close the short before the call time limit, or risk forfeiting their entire balance. To protect the short seller from always having to be online, we use an auction mechanism to allow third parties to place bids to sell back the original *base token*.

During the call period, if a short has not yet been closed, any third party can send a transaction containing an offered price at which to sell back the original *base token*. Upon receiving such a valid transaction the *ShortSell* contract will immediately take and hold onto the original amount of *base token* from the sender. The sender will not be sent the *quote token* indicated in their offer until after the call time expires. If the short seller closes the short sell in the traditional way, or a better offer is received, the *base token* will be sent back to the initial bidder. Only offers with a better price for the short seller will be accepted. It is in third parties' interest to submit these bids if they believe they can get a better than market exchange rate of *quote token* for their *base token*. It is likely the exchange rate received from this bidding process will not be quite as optimal as if the short seller chose an order himself as the bidders must pay gas costs, but it should still offer short sellers protection against their shorts being called when they are unable to close.

3.2.4.7 Additional Deposits

The loan will normally be called in by the short seller when he is concerned the *quote token* deposit locked in the *ShortSell* contract will not be enough to cover his interest fee and repurchase of the original loaned *base token*. In order to give himself more time before this happens the short seller can post more deposit in *quote token* as additional collateral for the loan. This is done by the short seller sending the *ShortSell* contract a transaction indicating he wishes to post additional deposit, which causes the indicated amount of *quote token* to be transferred from the short seller to the *ShortSell* contract. This can be done at any time before the short is closed. It is expected the lender will use off-blockchain communication or a set of rules to indicate when additional deposit must be posted or the loan will be called in, but this is not specified by dYdX.

3.2.5 Risks

One risk for the short seller is that the lender calls in the loan before the short seller wishes to close the short even when enough deposit is posted. Current non-blockchain related financial systems use a reputation system to identify optimal lenders that will not call in the loan prematurely. Such a reputation system for dYdX could exist entirely separate from the base protocol, as short sellers would prefer loan offers from lenders with higher ratings and would price this into their decision on whether or not to take a loan.

The risk for the lender (besides the economic risk of holding the *base token*) is that the price of the *base token* relative to the *quote token* rises so rapidly that he is not able to call in the loan before the amount of *quote token* locked in the *ShortSell* contract is no longer enough to buy back his original amount of *base token*. In this case the lender would still receive the entire amount of *quote token* locked in the short sell, but would have been better off just holding the *base token*. This risk for the lender can be mitigated by setting a high enough deposit, low enough call in time, and short enough lockout period on the loan.

4 Governance

To increase efficiency and promote a common standard, all derivative protocols offered by dYdX will be governed by a single decentralized entity. This governing entity will have the authority to upgrade certain upgradable contracts used by the dYdX protocol. Some examples of upgradable contracts in the previously described derivative protocols include the *Creator* in the covered call protocol, and the *ShortSell* contract in the short sell protocol. Upgradability is essential to the success of the protocol, as it must adapt to changes in the underlying platforms it uses as well as to changing market demands.

Governance will initially be handled by a multisig contract whose keys are held by reputable individuals with a vested interest in the success of dYdX.

In the future dYdX will upgrade to use a DAO to govern its protocols. No viable DAO currently exists, however extensive research is currently going into decentralized governance structures by several other projects. dYdX hopes to work together with leading projects to further research in this field, and come up with a solution to allow governance of the protocol by its users themselves.

5 Summary

- dYdX
  - Decentralized derivatives protocol
  - Built on Ethereum and 0x
  - Open and free to use
  - Efficient markets are enabled using off-chain 0x orders and economic incentives for price discovery
  - Decentralized governance and upgrade mechanism allows protocol contracts to be continuously improved
- Options
  - Can be used to reduce risk or speculate
  - Anyone can create, write, buy, or trade any option on any ERC20 token
  - Each option is represented by its own ERC20 token to allow easy trading
- Short Sells
  - Can be used to profit on downward price movements
  - Providing low risk fully collateralized loans for short sells can provide interest fee on long positions
  - Anyone can short or lend any ERC20 token

6 Acknowledgments

We would like to thank our mentors, advisors, and friends who have provided invaluable advice on dYdX. In particular, we would like to thank Olaf Carlson-Wee, Fred Ehrsam, Linda Xie, and Julian Borrey for reviewing and providing feedback on this work. We would also like to thank Albert Zhou for his help educating us about derivatives. dYdX would not have been possible without their and others' support.