MARKETPROTOCOL

WHITE PAPER

TABLE OF CONTENTS

SUMMARY

Currently, centralized exchanges have significant shortcomings including security, solvency, and custody of funds. They require substantial trust from Traders while providing limited transparency in return. Blockchain technology creates an opportunity for unique solutions to these problems by providing the transparency, security and trustless custody of funds necessary for a responsible trading ecosystem.

MARKET Protocol ("MARKET") has created open source foundation blocks that provides the interoperability necessary for building decentralized exchanges and conducting trading activities on the Ethereum blockchain. As an open-source protocol governed by its participants, MARKET does not levy any fees or extract value from the system. It provides the framework enabling Traders to buy and sell digital and real-world assets in a safe, solvent and trustless marketplace.

To further extend usability, Third party projects can build decentralized applications or "dApps" on top of the protocol, enabling non-technical Traders to interact with the underlying blockchain easily. The MARKET team is preparing to release the first dApp for public beta, which allows users to deploy MARKET Smart Contracts and explore those created by others.

MARKET Smart Contracts are similar to traditional derivatives in that they are a contract between two or more individuals, which settles in the future based on the price of an underlying asset. Traders create relationships like TSLA/ETH utilizing their digital assets as collateral for Tesla stock without converting to fiat currency. Traders deposit collateral funds into the MARKET Smart Contract before trading. At execution, the funds become locked to guarantee contract solvency.

The MARKET Protocol team comes from diverse technical and financial backgrounds with over 30 years of cumulative electronic trading experience on global exchanges. Co-founders Seth Rubin, Phil Elsasser, and Collins Brown have been working together since 2014 managing a 24-hour algorithmic trading group. These experiences enabled the team to see how blockchain could solve many of the problems inherent to traditional and crypto exchange models. These insights catalyzed the development of MARKET Protocol, to create an open, trustless, and decentralized trading marketplace.

PROBLEMS ADDRESSED BY MARKET

Traditional Derivatives Exchanges

1. **Margin Funding, Forced Liquidations and Leverage**
Contracts created through MARKET have a defined upside and downside. For example, If the underlying price moves by 50% (to the upside or downside, as defined by the contract creator) all positions are automatically settled, and the contract's life-cycle is complete.

A trader submits funds before trading a MARKET contract. At execution, the maximum downside of each trade is committed to the collateral pool until the position is exited.

Traditional exchanges regularly deal with margin calls or forced liquidations. MARKET removes the systemic risk of leverage from the environment allowing Traders to deploy capital efficiently and enter positions with a predefined downside, preventing dangerous and disruptive market liquidations. In times of market stress, forced liquidations tend to further cascade into additional liquidations.

For example, In June 2017, a large multimillion dollar sell order on GDAX caused the price of Ethereum to drop from $317.81 to $224.48 and then triggered a flood of 800 stop-loss and margin liquidation orders to further drop the price down to $0.1 only to recover minutes later.[1] The 2017 Ethereum Flash Crash is systemically dangerous. These market conditions could have been mitigated or prevented entirely by using MARKET contracts.

2. Clearing Firms and Custody of Funds
Centralized derivatives exchanges rely on clearing firms to handle customer funds. In 2011, MF Global reported a shortfall in customer accounts as large as $1.2 billion.[1] When MF Global declared bankruptcy, approximately 33,000 customers had their funds frozen.[2] Custody of funds and segregation of customer funds are concerns with current exchange models.

3. Market Solvency
By design, exchanges obfuscate counterparties and in doing so make it difficult to view the market as a whole. Participants must trust that clearing firms and exchanges are appropriately monitoring Trader's positions and capital balances. There is no way to guarantee market solvency.

4. Commercial Contracts
Exchanges, as for-profit entities, only offer contracts they believe will be commercially successful. They have minimum market participation and open interest goals for new contracts to attract more significant institutional clients. As a result, potentially useful

---

1 Bloomberg News. (2011). MF Global shortfall may be more than $1.2B. Available at:

2 Sauer, F. (2014). The MF Global Collapse Explained (And Why It Is a Crime). Available at:

contracts may not get listed, forcing Traders and businesses to transact only in contracts the exchanges choose to list.

5. Limited Access/Credit Verification
Existing exchanges require credit verification and minimum account balances restricting availability.

Cryptocurrency Exchanges

Cryptocurrency exchanges are different from derivatives exchanges and more closely resemble the US equity markets. Many fragmented cryptocurrency exchanges provide a venue for participants to engage in discrete transactions like trading Bitcoin ("BTC") for US dollars ("USD") or Ethereum ("ETH") for other ERC20 tokens.

Typically, the exchanges charge the user a fee expressed as a percentage of the trade or charges for withdrawals or fiat currency deposits. The fee can vary depending on the order type. Cryptocurrency prices differ from exchange to exchange due to the market participants' perception of each exchange's capital controls, solvency, liquidity, regulatory regime, and other factors.

Given the increasing worldwide user base for cryptocurrency exchanges and institutional demand, The Chicago Board Options Exchange, The CME Group, and LedgerX began trading BTC options and derivatives. However, these projects were built on top of traditional exchange and clearing models and will continue to perpetuate the problems discussed in prior sections. They also require Traders to post USD margin.

1. Digital IOUs
Trades on centralized crypto exchanges such as Coinbase and Kraken do not result in actual transactions on the blockchain; instead, they are internal transactions on the exchange's ledger. Only when a user withdraws crypto tokens from an exchange to the user's digital wallet does a blockchain transaction take place, thus giving users control over the tokens. Until that point, funds reside in a commingled exchange wallet leaving Trader's funds at risk.

2. Limited Products
Limited products exist and proposed decentralized exchanges limit trading to ERC20 tokens with no ability to trade the majority of other cryptocurrencies, including Bitcoin, Litecoin or Ripple. Additionally, all trades are A for B, atomic transactions. Traders can only trade assets they own. If they want price exposure to multiple crypto assets, they must buy, hold and store each one.

3. Cannot Short
Traders have limited options to short crypto assets or bet on a decline in price. Currently, there is no easy way for Traders to borrow or sell assets and shorting crypto assets for the average Trader is complicated.

4. Inefficient Markets

Short sellers provide an essential force in the marketplace by allowing speculators and hedgers to bet on a decrease in price. Without short sellers, a fundamental piece of an efficient market is missing. Further, most Traders, including U.S. based Traders, cannot deploy leverage on crypto assets and are thus forced to post the entire value of their open position resulting in inefficient capital allocations.

MARKET PROTOCOL

MARKET provides developers with a trustless and secure framework to create decentralized exchanges, including the necessary clearing and collateral pool infrastructure. As a protocol, MARKET enables third parties to build applications for trading, order routing and related activities.

The decentralized protocol facilitates risk transference and a trustless trading system through smart contracts on the Ethereum blockchain. MARKET contracts derive their price from an underlying asset, either digital or real-world. Traders are not limited to owned or existing ERC20 tokens, allowing price exposure to other cryptocurrencies like Bitcoin, Ripple, and Monero.

As derivatives, MARKET contracts offer users continuous price exposure and future settlement. Traders can quickly enter long or short positions in any contract where they find liquidity.
Trade participants then contribute funds to a collateral pool before trade execution. The contract then distributes funds in a rule-based manner at an agreed-upon settlement date or when Traders exit positions before the settlement date.

The clearing functionality provides a safe and secure framework to manage crypto assets, positions, and leverage in a systemically responsible way. All smart contracts and collateral pool balances are publicly available on the blockchain. No person or entity controls the flow of assets among participants, order matching, contract creation or dispute resolution.

Participants will govern the protocol in a democratic and equitable fashion. Traders of the protocol will be the owners and decision makers. The goal of MARKET is to provide users the most efficient, safe, and secure environment possible while creating a robust and fair marketplace.

TECHNICAL SPECIFICATIONS

Overview

MARKET allows third parties to create “markets” by hosting an order book. The order book hosts, referred to hereafter as “nodes”, are incentivized to host order books by collecting transaction fees, which they set and control. MARKET simplifies the complexity of securing collateral, validating creditworthiness, executing settlement and the custody of customer funds with smart contracts.

At this time, nodes are not responsible for the matching of trades and never have custody of funds. Traders of the protocol can post orders as makers, or they can trade against resting orders as takers. Nodes act in a bulletin board like fashion and broadcast all maker orders transmitted to them providing potential matches to takers, who ultimately select and execute the trade

In the future, MARKET may pursue alternative node solutions, such as a fully decentralized order book and matching, as well as other scaling implementations.

Contract Creation and Clearing

MARKET allows users to create a contract, specify its terms, publish those terms, and provide a mechanism for automated settlement while ensuring contract solvency through collateral pools. Any Trader can create a new contract by outlining the contract specifications. The contract creator will be presented with the following options:

- **Underlying Instrument:** What is the underlying asset for pricing? This could be a real world or digital asset.
- **Price Floor and Cap:** This defines the maximum loss or gain for participants and the amount of collateral each participant must post in order to take a short or long position.
- **Expiration Date**
- **Settlement mechanism:** Creators select an oracle-based solution for the final settlement price of the contract used in profit and loss calculations
- **Base Token:** What will the base currency be for pricing? This dictates the ERC20 token contributed to the collateral pool

Since the value of the contract is derived from the value of a held cryptocurrency, contracts can be created for any physical asset, digital asset, or ERC20 token.

Shared Collateral Pool

Each MARKET contract is comprised of multiple smart contracts that create the shared collateral pool and needed accounting of an individual Traders’ balances denoted in the defined base token (any ERC20).

Traders will deposit collateral in the form of ERC20-compatible tokens to the smart contract prior to trading, and all profits and losses from trades will be settled using the tokens.

After depositing tokens, the Trader can submit orders and enter positions based on their smart contract balance. When a Trader opens a position, tokens will be transferred from the Trader's balance to the collateral pool. The tokens in the collateral pool fully fund the max loss of all open positions within a specific contract. If a Trader closes a position prior to settlement, their previously allocated capital - plus or minus any profit or loss - is available for withdrawal or further trading. Alternatively, a Trader can hold their position through expiration. In that case, a oracle will provide a settlement value which is used to determine the Trader's profit or loss. Once the contract enters a settled state, users may call a function to return their collateral.

All open positions are fully collateralized at the time of execution removing counterparty risk and replacing one of the core functionalities of traditional exchanges. A smart contract governing the collateral pool will provide a reliable and trustless solution to traditional custody of funds issues.

The executed trade price and quantity determine the amount of collateral moved from the Trader's balance to the collateral pool. Allocated collateral equals the maximum loss possible for that position. For buyers, it is the entry price minus the contract minimum, and for sellers, it's the contract maximum minus the entry price.

```
    /// @notice determines the amount of needed collateral for a given position (qty and price)
    /// @param priceFloor lowest price the contract is allowed to trade before expiration
    /// @param priceCap highest price the contract is allowed to trade before expiration
    /// @param qtyDecimalPlaces number of decimal places in traded quantity.
    /// @param qty signed integer corresponding to the traded quantity
    /// @param price of the trade
    function calculateNeededCollateral(
        uint priceFloor,
        uint priceCap,
        uint qtyDecimalPlaces,
        int qty,
        uint price
    ) pure internal returns (uint neededCollateral) {

        uint maxLoss;
        if(qty > 0) {   // this qty is long, calculate max loss from entry price to floor
            if(price <= priceFloor) {
                maxLoss = 0;
            }
            else {
                maxLoss = subtract(price, priceFloor);
            }
        } else {        // this qty is short, calculate max loss from entry price to ceiling;
            if(price >= priceCap){
                maxLoss = 0;
            } else {
                maxLoss = subtract(priceCap, price);
            }
        }
        neededCollateral = maxLoss * abs(qty) * qtyDecimalPlaces;
    }
}
```

This collateral remains in the pool until the trade is closed. Next, the contract updates the price and quantity of the user's open positions.

Leverage and Contract Range

MARKET contracts offer continuous profit and loss exposure derived from an underlying asset up to a PRICE_CAP and PRICE_FLOOR specified during contract creation defining the contract range. Leverage offered through the MARKET protocol differs from traditional leverage, which runs the risk of forced liquidations and unfunded positions.

Traders will commit the difference between the executed price and their maximum loss to the collateral pool when they initiate a new trade. This action will require less equity than the total notional value of the position providing implicit leverage. All prices between initial entry within the contract range are tradeable. The outcomes are not binary.

If the high or low of the range is breached the contracts is settled with the participants on one side awarded their maximum gain, while the other side receives nothing (their maximum loss). It is possible neither is breached, in that case the contract trades and expires conventionally. We plan to implement this functionality to ensure that the market remains solvent. *This process is one of the most important features of the contract framework and the MARKET protocol.* The amount of leverage afforded to an open position depends on where the price of the trade executed relative to the ranges of the current contract. For example, users will post less margin when selling near the maximum as they have less downside.

Traders can replicate uncapped payoff structures by stripping together a series of contracts. We expect third-party implementations of MARKET to provide multiple strikes per contract and an easy, cost-effective way for Traders to create the exposure they want. This may present Traders with arbitrage opportunities as Traders can spread-trade multiple contracts against each other.

Two detailed examples of trading the MARKET protocol are including later in the document.

Short Selling

Currently, there are limited and inefficient options to short crypto assets. However, a MARKET contract makes shorting simple.

If two parties are willing to transact at a predefined price they can trade. There is no need for the short to locate or borrow the underlying asset. With MARKET, if a contract is listed and has liquidity, it can be shorted.

Order Submission and Execution

To begin trading, a user will first commit the requisite amount of the base token to the collateral pool smart contract, thus ensuring funds are available to trade. Funding the smart

contract prior to execution results in fewer transactional failures during matching and creates a better user experience.

For a Trader to enter a trade, they will submit an order, as a maker, to a node providing both a price and quantity. Upon receipt, the node confirms that the maker's address has the necessary balance in the smart contract to place the order. Next, the node will post and maintain the order in the order book until another Trader (taker) fills it. For providing this service, the node sets and collects a transaction fee. The taker is responsible for filling the maker order by calling the trade function via a smart contract and supplying their address. The taker controls order matching, reinforcing the trustless role of the node. At that point, funds are moved from the Trader's smart contract balance to the collateral pool. The node never handles funds. Finally, the new positions for each participant are recorded in the smart contract and on the blockchain.

```
struct Order {
    address maker;
    address taker;
    address feeRecipient;
    uint makerFee;
    uint takerFee;
    uint price;
    uint expirationTimeStamp;
    int qty;
    bytes32 orderHash;
}
```

If multiple executions exist, positions are exited in a LIFO (Last In, First Out) manner. After exiting a position, the appropriate amount of collateral (including any gain or loss) will be allocated back to the user's smart contract balance and become available for trading or withdrawal.

```
struct UserNetPosition {
    address userAddress;
    Position[] positions;   // all open positions (lifo upon exit - allows us to not reindex array!)
    int netPosition;        // net position across all prices / executions
}

struct Position {
    uint price;
    int qty;
}

```

Expiration and settlement

Upon the expiration of a contract, functionality built into MARKET allows contracts to be settled using an oracle such as Oraclize.it or Thomson Reuters Block1IQ. Oracles provide external data to the blockchain. The contract creator will have the ability to set the frequency for oracle queries.

```
    /// @param queryID of the returning query, this should match our own internal mapping
    /// @param result query to be processed
    /// @param proof result proof
    function __callback(bytes32 queryID, string result, bytes proof) public {
        require(validQueryIDs[queryID]);
        require(msg.sender == oraclize_cbAddress());
        lastPriceQueryResult = result;
        lastPrice = parseInt(result, PRICE_DECIMAL_PLACES);
        UpdatedLastPrice(result);
        delete validQueryIDs[queryID];
        checkSettlement();
        if (!isSettled) {
            queryOracle();  // set up our next query
        }
    }
```

Typically, the oracle is queried once a day to determine if a contracts price bands have been exceeded or if the contract is past the expiration date. If either of these criteria are met, the contract enters an expired state, and the settlement process begins. Additionally, a user may call a function at any time to induce settlement if a settlement condition is met, such as, an exceeded price band.

```
    /// @dev checks our last query price to see if our contract should enter settlement due to it being past our
    //  expiration date or outside of our tradeable ranges.
    function checkSettlement() private {
        if(isSettled)   // already settled.
            return;

        if(now > EXPIRATION) {  // note: miners can cheat this by small increments of time (minutes, not hours)
            isSettled = true;   // time based expiration has occurred.
        } else if(lastPrice >= PRICE_CAP || lastPrice <= PRICE_FLOOR) {
            isSettled = true;   // we have breached/touched our pricing bands
        }

        if(isSettled) {
            settleContract(lastPrice);
        }
    }
```

Contracts can be settled to the price of any actively traded ERC20 token, crypto currency or other listed asset by calling an exchange API via an oracle, e.g., the contract defined settlement procedure could specify the last traded price of a token on Kraken at a predetermined time.

To avoid incorrect or inaccurate settlement prices, we will implement a time delay between the initial execution (contract expiration) and the time at which users may withdraw their funds. If more than a certain percentage of the participants with open positions initiate a settlement dispute, then the contract enters a disputed state.

In the event of a settlement dispute, a backup oracle or group of backup oracles can be used to obtain a settlement value. As crowd based consensus mechanisms evolve MARKET intends to implement additional resolution mechanisms. Until that point, disputed settlements may also get resolved through a more centralized process to ensure funds are equitably returned to participants and not permanently trapped in the contract.

DAPPS

In order to get the most out of MARKET for the non-technical user, distributed apps or "dApps" will be built by MARKET for use on the MARKET Protocol in addition to third party developers. MARKET plans to create simple user interfaces that intuitively explain the process of selecting contract variables and deploying MARKET contracts to the blockchain. Additionally, a contract explorer will provide the ability to search previously deployed MARKET contracts and their specifications. Users will also be able to test their oracle queries to ensure they function as expected prior to contract deployment.

MKT TOKEN

MKT will be the base token of the MARKET ecosystem and benefits from integration into all facets of MARKET. Peer-to-peer trading is free on MARKET and no fees are native to the protocol. Nodes providing order book hosting and management will have the option to set and collect transaction fees for offering this service. Orders submitted without the required transaction fee may be rejected by the node. These transaction fees will be collected by the nodes in MKT.
MKT holders will also have a vote in protocol improvements and development. This ensures both users and projects using MARKET Protocol have a voice in the protocol's future.

Token Use

1. **Contract Access:** Initially, participants are required to post 25 MKT tokens along with the appropriate base token (for collateral) to trade each user-defined contract. MKT posted in this way is returned to the user when they stop trading a specific contract or upon expiration.

2. **Transaction fees:** Nodes provide a service to users of MARKET Protocol and in exchange for this service, may charge a transaction fee denominated in MKT. Each node sets its own fee for the service.. Nodes are expected to differentiate themselves on the fees and services provided giving users many choices for execution.

3. **Settlement:** For accuracy and autonomy, most contracts will automatically settle to publicly accessible oracle solutions. In the event of a settlement disagreement, or a disrupted settlement process, MARKET intends to a employ number of solutions including, backup oracles or a crowd sourced resolution pulled from the pool of MKT holders.

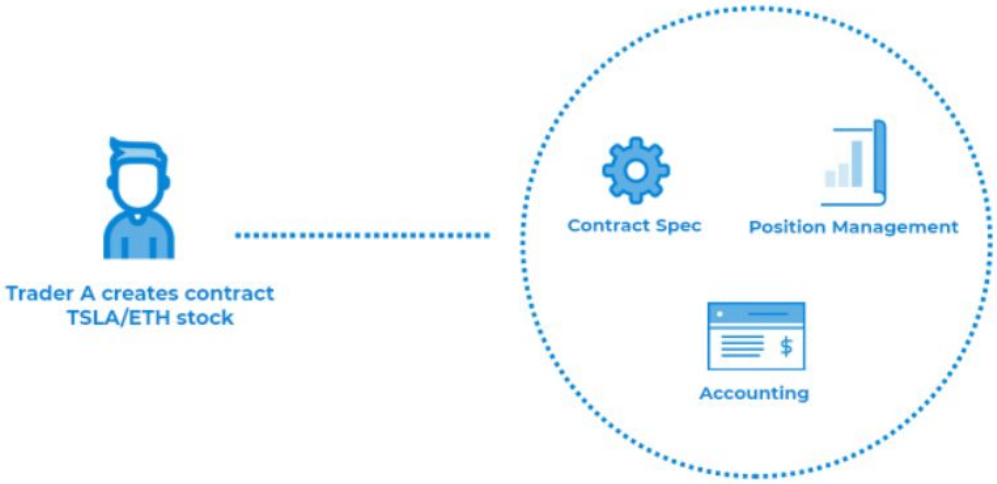
4. **Contract Creation:** Initially, users will be required to hold a minimum of 500 MKT to create a new trading contract. The purpose of this arbitrary holding is to encourage thoughtful contract creation.

5. **Protocol Decisions:** We expect MKT token holders to vote on protocol decisions and development. This will include things like number of MKT tokens required for contract access or creation as indicated above. The more MKT an account holds; the more influence it will have. At the onset, decision making will be more centralized. MKT holders represent the whole ecosystem, traders, nodes and speculators and each has a voice in protocol direction. Over time, our goal is to move towards a more community-based decision model.

MKT CONTRACT EXAMPLES

Example A – Single Stock Contract

In this example, we showcase how the MARKET protocol can be used to create a derivative contract between ETH and a traditional security, Tesla stock (NASDAQ, TSLA)



User A will define:

1. A base currency (any ERC20 token, in this case ERC20 compliant ETH)
2. An underlying asset (in this case TSLA)
3. Settlement (with oracle)
4. CAP and FLOOR- If the price of TSLA/ETH is currently 10, the contract may be created with a CAP of 15 and FLOOR or 5. If the contract underlying instrument goes to 5 or 15 it expires. All prices are tradeable between 5 and 15.

At expiration all users are paid out their gain or loss.

Pre-Trade

1. The Trader identifies a contract spec they want to trade. In this case it is TSLA/ETH
2. Trader A deposits 10 ETH and Trader B 7 ETH to the smart contract to begin trading. They also deposit 25 MKT each.
3. Users with balances can withdraw their balance at any time. Funds committed to open positions are not eligible for withdrawal.
4. Contracts have a shared collateral pool used to hold funds for open orders and positions.

TSLA/ETH initial balances

Placing an Order

1. Trader A creates an order object to buy 1 TSLA/ETH at price 10, signs it and transmits the order to the node.
2. The defined TSLA/ETH contract has a CAP of 15 and a FLOOR of 5.

3. Node confirms that Trader A has funds available in smart contract to create this

order.

4. Node accepts order, and displays a bid for 1 TSLA/ETH at price 10.

Trade Execution

1. User B looks at the orders held by the node sees user A's order for 1 bid at price 10. User B wants to sell 1 ETH/Tesla at price 10
2. User B then takes User A's order calling the fill trade function to the MARKET smart contract with the order information
3. The MARKET smart contract then fills the order and allocates positions.
4. Collateral balances and user balances are updated. Each user has their max loss added to the collateral pool
5. Transaction fees designated for the nodes (in MKT tokens) are sent at execution.

Post-Trade

Using an TSLA/ETH contract with a CAP of 15 and FLOOR of 5 there are three post trade scenarios. If either 15 or 5 is traded in the **underlying** TSLA/ETH pair, then the contract expires and goes to settlement

1. Traders exit position prior to expiration
2. Traders hold position to expiration
   Contract price CAP or FLOOR is breached

Scenario 1: Users exit positions prior to expiration

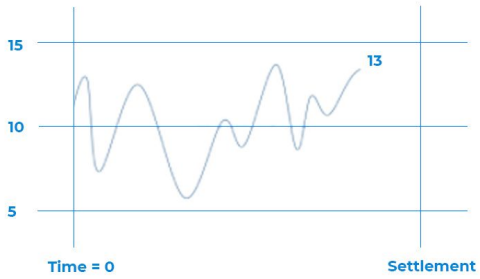
Both Traders have an open position, A is long 1 at 10 and B is short 1 at 10. The TSLA/ETH relationship which was trading at 10 is now trading at 13.

Trader B wants to realize a loss and creates a new order in the same process indicated previously and submits the order to a node. As a closing order, there is no need for additional collateral.

Trader A wants to realize a profit and fills User B's order.

Both traders are flat. Trader A has made 3 ETH and Trader B has lost 3 ETH.

Initial collateral balances were 5 ETH each. Trader A receives back 8 ETH and Trader B 2 ETH. Total balance of pool was 10 ETH and is now 0. If they are done trading this contract, both receive back their MKT tokens.

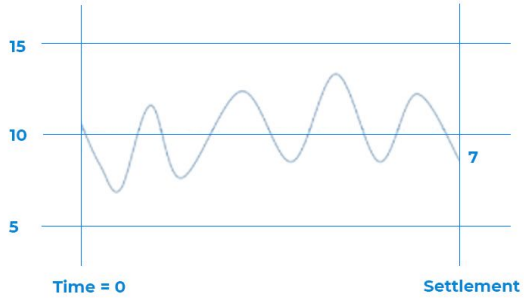


Scenario 2: Traders hold trade till expiration - 1 month from contract creation

Both users have an open position, A is long 1 at 10 and B is short 1 at 10.
The contract organically settles. An oracle delivers the necessary settlement price based on the contract specification which is is used to determine the Traders' PNL.

In this case, the contract settles at 7.

User A has lost 3 ETH and receives back 2 ETH (initial deposit + PNL) and User B made 3 ETH and receives back 8 ETH (initial deposit + PNL).

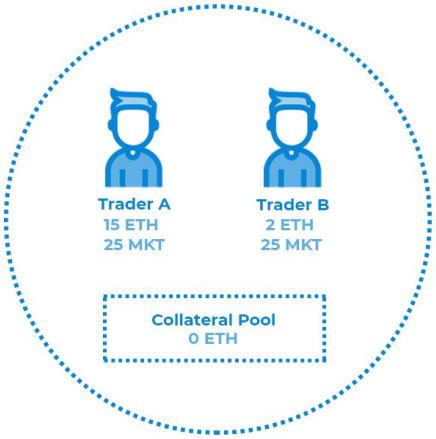


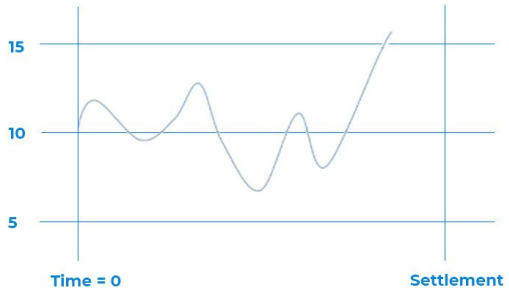
Scenario 3: Contract Hits CAP of 15

Both traders have an open position, A is long 1 at 10 and B is short 1 at 10.

In this case, the contract hits the upper bound trading 15 When this happens the contract automatically expires at a price of 15.

Trader A made 5 ETH and receives back 10 ETH (initial deposit +/- PNL) and Trader B lost 5 ETH and receives back 0 ETH (initial deposit +/- PNL).





All open positions are recorded on the blockchain and transparent to all users. The balance of the collateral pool is always fully funded to cover all open positions. As a user trades out of open positions, the accounting is done in a last in first out method.

Example B – Hedging a Utility Token

Users can hedge utility tokens with MARKET removing price movement both up and down. The majority of existing and future ICO tokens provide the owner some benefit or utility. These tokens, however, may have considerable price volatility which could actually outweigh any potential benefit associated with the token. MARKET provides owners of utility tokens a way to hedge their price exposure while maintaining the utility associated with owning the tokens. Token owners never sell or transfer their tokens.

In this example, we will use SALT lending tokens. SALT is a peer to peer lending platform allowing users to borrow fiat through loans backed by crypto holdings. SALT Lending tokens are necessary to participate on the platform and obtain loans.

Since they were issued, SALT tokens have traded from the low $2s to a high of over $17. With MARKET Traders can hedge this price volatility.

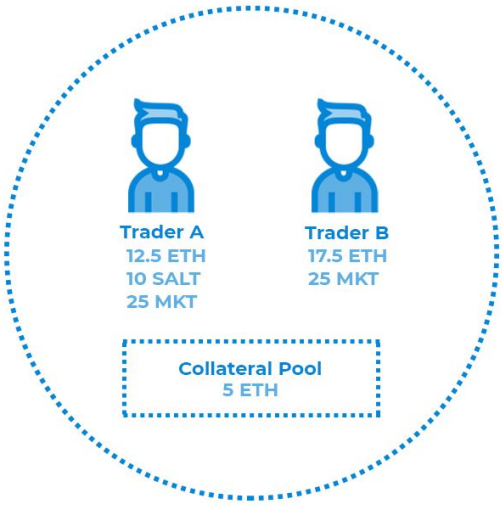
To illustrate this example, suppose Trader A owns a number of SALT tokens and wants to hedge their price exposure:

Pre-Trade

1. Trader A owns 10 SALT tokens worth 5 ETH and wants to hedge them, he would need to sell 10 contracts.

2. Trader B doesn't own SALT tokens but wants to speculate on the price of SALT tokens

It is important to note, that Trader A still has his SALT tokens and can use them to access the platform. Trader B never had SALT tokens and still doesn't.



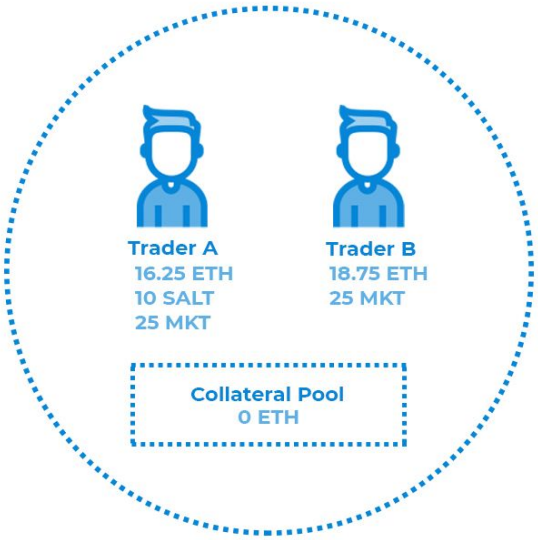
For Trader A to hedge his SALT token price exposure, he would need to sell 10 contracts. The current price is 0.5 ETH. This means they have a maximum downside of 0.25 ETH * 10 = 2.5 ETH
Trader B is willing to buy SALT/ETH at the same price. From here SALT tokens can go up or down.

Post Trade

We will cover two post trade scenarios:

1. SALT goes down.
2. SALT goes up.

Scenario 1: SALT goes down from 0.5 to 0.375 and Traders exit positions.



Trader A gains 0 125 * 10 contracts = + 1.25 ETH

Trader B loses 0 .125 * 10 contracts = - 1.25 ETH

Trader balances are updated.

Trader A's original SALT tokens were worth 5 ETH

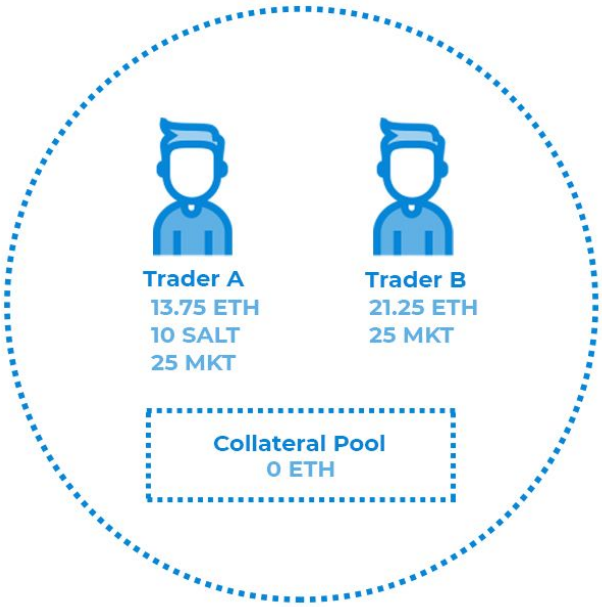
(0.5 * 10), they have since decreased in value to 3.75 ETH (0.375 * 10).

Trader A gained 1.25 on the SALT/ETH contract but lost 1.25 on his actual token holdings offsetting the loss.

If Trader A did not hedge his holdings with MARKET they would be exposed to the loss.

Trader B never owned any SALT tokens but through the SALT/ETH MARKET contract bought the equivalent of 10 tokens which decreased from 0.5 ETH to 0.375 ETH and provided a loss of 1.25 ETH. This which is the same as if they owned 10 SALT tokens without ever touching the original tokens.

Scenario 2: SALT goes up from 0.5 to 0.625 and Traders exit positions.



Trader A loses 0.125 * 10 contracts = 1.25 ETH

Trader B gains 0.125 * 10 contracts = 1.25 ETH

Trader balances are updated.

Trader A's original SALT tokens were worth 5 ETH

(0.5 * 10), they have since increased in value to 6.25 ETH (0.625 * 10).

Trader A lost 1.25 on the SALT/ETH contract but gained 1.25 on his actual token holdings offsetting the loss.

Trader B never owned any SALT tokens but through the SALT/ETH MARKET contract bought the equivalent of 10 tokens which increased from 0.5 ETH to 0.625 ETH and provided a gain of 1.25 ETH which is the same as if they owned 10 SALT tokens without ever touching the original tokens.

TEAM

The MARKET Protocol team comes from diverse technical and financial backgrounds with over 30 years of cumulative electronic trading experience on global exchanges. Co-founders Seth Rubin, Phil Elsasser, and Collins Brown have been working together since 2014 managing a 24-hour algorithmic trading group and started trading cryptocurrencies the next year. These experiences enabled the team to see how blockchain could solve many of the problems inherent to traditional and crypto exchange models. These insights catalyzed the development of MARKET Protocol, to create an open, trustless, and decentralized trading marketplace.

Seth Rubin
Co-Founder, CEO
seth@marketprotocol.io
Linkedin

Seth is involved in execution and development of the practical strategies incorporated into the MARKET protocol. Since starting his career as a derivatives trader in 2005, he has grown and managed multiple algorithmic trading desks, operated as a registered market maker and participated in numerous product launches. He was first introduced to the crypto space in 2015 as a Trader and focused later on the technology side. Seth, in conjunction with Collins and Phil, has developed and successfully implemented arbitrage and relative value crypto strategies. Seth has a strong understanding of centralized and decentralized trading and exchange infrastructures.

Collins Brown
Co-Founder
collins@marketprotocol.io
Linkedin

Collins began his career at Transmarket Group and has almost 13 years of experience trading various asset classes on global exchanges. He co-founded BRE Trading in 2014. As a career Trader, he understands the demands of an exchange user. Collins believes it is important that MARKET embraces the strength and weaknesses of both centralized and decentralized exchanges.

He has managed successful development groups solving complex problems. Collins has directed crypto market microstructure trade and research projects. He also has experience testing algorithms and software in QA and production.

Phil Elsasser
Co-Founder, CTO
phil@marketprotocol.io
Linkedin

Phil has spent the last 7 years as the lead developer on a algorithmic trading desk. Phil has led teams of developers focused on creating trading infrastructure, user interfaces, execution platforms and quantitative trading analytics. Phil has the skill-set, creativity and passion needed to implement technical solutions to solve the challenges presented by a decentralized market place.

He has written low-latency connectivity and trading strategies to futures exchanges like the CME, ICE, TOCOM, SGX and others. Leveraging his experience in the traditional derivatives space, Phil has also written strategies to a number of crypto exchanges.

Phil is currently a technical and security advisor to a crypto assets hedge fund.

Lazar Jovanovic
Marketing / Brand Ambassador
lazar@marketprotocol.io
Linkedin

As crypto trader and enthusiast Lazar has evaluated many new projects, pursuing the technical side of more successful ones. He has been involved with development strategy and community support for a number of blockchain based startups.

Mauzy Keshavarzi
Community Manager
maz@marketprotocol.io

Maz has acted as a business consultant helping a number of startups. He has experience developing and implementing successful online community outreach programs. He joined the MARKET Protocol team to combine his experience in crypto trading and asset management with his passion for helping others. Maz hopes to help the community learn what MARKET is all about.

Perfect Makanju
Software Developer
perfect@marketprotocol.io
Linkedin

Perfect is passionate about building smart solutions related to Android, AI and Smart Contracts in order to make people's lives better. He also has experience with machine learning, image processing and web applications (backend with php and nodejs)

Eswara Sai
Software Developer
eswara@marketprotocol.io
Linkedin

Eswara Sai is a passionate Frontend Developer proficient working with HTML, CSS & JavaScript. He is also experienced with JavaScript frameworks such as ReactJS and AngularJS which are majorly used in building dynamic single-page web applications. He works closely with Phil Elsasser and the rest of the team to build a Dapp for the MARKET to be used for deploying and testing the MARKET Contracts.

Przemyslaw Szulczynski
Software Developer
przemyslaw@marketprotocol.io
Linkedin

When Przemyslaw was 9 years old his father bought him Commodore 64 with a 5.25'' floppy drive. There were not many books around on the subject at that time. The only book he had was Commodore 64 Memory Map, which he read several times. After he got his first 90 Mhz PC, Przemyslaw spent two years on the Internet without the Internet connection. He was browsing offline mirror copies of the websites. Hyperlinks sometimes worked as expected.

Years later and many projects later he is still a technology enthusiast. Worked with great teams in multinational environments. Has an in-depth knowledge of Digital TV, MPEG2/H.264, HDCP, embedded devices, navigation devices, advanced routing algorithms and automotive projects that include bleeding edge hypervisors. Translated a book. Wrote several articles on MQL5. Interested in blockchains and derivatives trading. Believes that MARKET Protocol will have a bright future.

ADVISORS

Patrick Charles
Data Science and Analytics Pipeline Architect
Linkedin

Patrick Charles has over twenty years of experience building software in a variety of industries including finance, education, health care and computer security. He has worked as a technology leader, consultant, software architect, engineer and researcher. Patrick is an open source contributor, has authored a number of technical papers, is an inventor with two US patents and is co-author of the opening chapter in the soon to be published book “Frontiers of Cyberlearning”.

Josh Fraser
Co-Founder, Origin Protocol
Linkedin

Josh started coding at the age of 10. Prior to Origin, he co-founded three other venture-backed companies: EventVue, Torbit (acquired by Walmart Labs) & Forage.

Dan Horowitz
Senior Vice President Engineering, 1010data
Linkedin

As VP of Engineering, Dan oversees systems development for 1010data. He focuses on making 1010data the fastest, most reliable big data discovery and sharing platform on the web. He has been instrumental developing scaled solutions to gracefully manage petabytes of data with world-class reliability and fast query response time. During his 8-year tenure at 1010data, Dan has worked on numerous critical development projects across the company including the Trillion-Row Spreadsheet and various MBS data products. Before joining 1010data, Dan was a developer at Accenture on the Global Architecture Team building custom enterprise management software. Dan holds a BS in Computer Science from the University of Rochester.

Brian Shields
Co-Founder, Coder Inc.
Linkedin

Brian has over a decade of experience in debt and capital markets, digital marketing, and technology entrepreneurship. A recognized innovator, he has a wealth of business acumen developed through years of working with Fortune 500 and Inc. 500 companies, as well as early-stage startups. Brian acts as the venture mind for Coder, leading sales, marketing, finance and strategy. Before founding Coder, Brian was an Associate Director at Brafton, a digital marketing agency with offices in Boston, Chicago, and San Francisco. Brian graduated with a Bachelor's degree in finance from the University of Illinois at Urbana-Champaign.

Brent Traidman
Chief Revenue Officer, Bread
Linkedin

Brent has over 15 years of experience leading high impact growth software companies, of which many have had successful exits. Brent is currently the Chief Revenue Officer at Bread (BRD), one of the world's fastest growing crypto financial platforms. Bread is considered a thought leader in the world of crypto, with over one million users in over 140 countries. Prior to Bread, Brent worked in the Vista Equity Partners portfolio where he helped drive two exits. Brent builds revenue engines and helps develop growth strategies. When not at work, he helps advise a Silicon Valley venture capital firm on early stage investments, and regularly speaks, mentors, and attends global accelerator conferences.