KSOX
Decentralized Exchange and Payment Processor powered by zk-STARKs

Paweł Nowak

Abstract

KSOX is a modern platform based on the state-of-the-art cryptographic and web technologies, designed for cryptocurrency exchange and providing users with a pleasant and intuitive interface where they can conduct transactions and send transaction requests. The application offers a range of automatic currency conversion mechanisms, enabling clients to accept transfers in any cryptocurrency and pay for transfers in any cryptocurrency, often different from the other party's currency. The application assumes all exchange risks.

The API we provide is straightforward, and any company wishing to start accepting cryptocurrencies as a form of payment is able to connect under transparent terms.

A key aspect is that all our clients do not have to trust us as a company in any way. At no point do we become the owners of the clients' funds. The funds are not transferred to us, but to a smart contract that operates on networks such as Ethereum. We have no access to the clients' funds and cannot manage them. The only possibility to transfer funds from a client's account to another account is after obtaining the cryptographic signature of the transaction using their private key, to which we have no access.

1 Introduction

Cryptocurrencies were created to enable electronic transactions without intermediaries. Before the advent of cryptocurrencies, the only option for transactions without an intermediary was cash transactions. Banks, by introducing electronic transactions, became intermediaries. A high degree of trust in the banking service provider is required to enjoy the convenience of electronic payments, as we entrust our funds to that company. The emergence of cryptocurrencies proved to be a revolution in this regard. Unfortunately, micropayments in the world of cryptocurrencies have become rather expensive and impractical. Cryptocurrency exchanges started providing micropayment services at the cost of re-centralization. Now, just like banks, clients can entrust their funds to exchanges and must place trust in a given exchange. We know of many cases where exchanges' funds were illegally seized and lost, or the owners succumbed to the temptation of speculation or creative accounting with client capital, having complete access to their funds.

However, thanks to modern zk-stark technology, our platform can function without direct access to client funds. The only transactions we can perform are those in line with their intentions, as each transaction is signed by the user. Our smart contract accepts only those calculation results that are consistent with the client's signature, making it impossible for us to use the client's funds in an unauthorized manner.

2 Problems

Blockchain ensures data integrity and consensus at the cost of speed and scalability. In other words, it operates slowly, but its results are considered reliable. Unfortunately, many services require high throughput and cannot wait for a block to be accepted by the network. Additionally, requiring the network to accept a single transaction is very costly. A much better idea is to aggregate transactions into a group and send a confirmation of the correct execution of the transactions and the result of the calculation to the network. This is the essence of proof of computation integrity technology. It allows us to be an exchange and transaction platform while not having any rights to clients' funds.

3 Technology

Let's start by explaining how CEXs work. Traditional exchanges are essentially interfaces to a database that store the current states of clients' accounts, pending orders, and a mechanism capable of evaluating whether a given client's order with specific parameters, such as price, asset to be sold, and asset to be purchased, can be fulfilled considering the already pending orders. In our case, this module is written in Rust. We have tried other languages before, but none of them provide the same level of security and scalability as Rust, in our opinion. The entire exchange engine and user API are written in this language. The user-side application is written in the modern framework, SolidJS, which is currently the fastest framework for creating frontend applications. Let's move on to the characteristics of DEXs. In simple terms, the mechanism ensures that the user trading on a DEX remains in control of their funds at all times. In contrast, with CEXs, the company responsible for a given CEX becomes the manager of the client's funds when the client transfers their funds to the exchange. Cairo is the first provable programming language developed by Starkware. We use code written in Cairo to prove the integrity of calculations performed on our servers so that the smart contract can trust them. The calculations prove compliance with the client's intentions, and only in such cases does the contract accept the changes. The signature is verified, and a commitment scheme is implemented. The possibilities of non-deterministic calculations in Cairo are utilized at this stage. The provable program only receives input on which calculations to perform, while the program itself verifies the correctness and compliance with the user's intentions.

4 Architecture

We will present the architecture from the perspective of a user query, but first, let us explain what the user must do to start using the platform.

1. When logging in for the first time, the user provides a public key or a verifiable derivative of the public key.

2. The server, upon receiving the user data, first verifies if the user is who they claim to be. It sends random data to the user to be signed and, if the client signs correctly and responds, the user is logged in.

3. The client is assigned an ID for a vault within the smart contract where they can transfer their funds.

4. The user can now invoke the appropriate function on the smart contract, which will withdraw their funds and record the transaction. It is important to note here that the user transfers the funds to the smart contract, not the exchange wallet, so even though the funds have been transferred, we, as an exchange, do not have the power to manage those funds.

5. From this point on, the funds are available in the user's panel on our platform and directly within the smart contract.

Now let's see what happens when the user wants to place an order:

1. The user decides to place a limit order on our platform.

2. By clicking the submit button in our platform's application, the order data is sent to the server.

3. The server directs the order to the exchange engine, which decides whether to execute the order or add it to the list of pending orders.

4. Assuming the order can be executed, a transaction is created

5. The transaction, along with other transactions, is sent to the program written in Cairo, which is executed to verify the correctness of the transactions, check the clients' signatures on the orders, and ensure that the transaction meets the order's requirements (i.e., it is executed at the appropriate price).

6. A proof is produced along with the program's output, which is sent to the smart contract to update the state.

7. The smart contract verifies the proof, and if it is correct, updates its state according to the operation results.

8. Additional data is sent to the smart contract to ensure that the correct data is provided as input to the provable program and that the commitment scheme is implemented.

In this way, the client completes a transaction. At no point do we become managers of the funds, so we cannot execute any operations other than those consistent with the user's intentions.

- The matching engine and server are written in Rust.
- The entire architecture, except for the database and smart contracts, is stateless, allowing for easy scaling of the application.
- Smart contracts are mainly written in Solidity (for EVM) but also in other languages designated for specific networks.
- We utilize non-deterministic calculations in the provable language. Cairo receives output from the matching engine and verifies the correctness of executed orders based on this information.
- Orders in the database are represented as a graph, specifically a directed multigraph where vertices correspond to assets and edges carry information about pending orders.
- Thanks to an elegant mathematical model, the matching engine's code is much less prone to errors.
- An AI-supported payment processor is specially trained for optimizing exchange costs.

5 Business Model

Exchange

A cryptocurrency exchange is an online platform that facilitates the buying, selling, and trading of various cryptocurrencies. The primary source of revenue for the exchange is through fees charged on transactions and services provided. The exchange will also need to do arbitrage to other exchanges to decrease the spread and keep the exchange rate as close to the market rate as possible.
The main sources of revenue are listed below:

- Transaction fees
- Arbitrage to other exchanges

Payment Processor

A cryptocurrency payment processor is a service that enables merchants to accept cryptocurrencies as a form of payment for their products or services. By integrating the payment processor, merchants can streamline transactions, reduce fraud, and tap into a new customer base. The primary source of revenue for the payment processor is through transaction fees and value-added services. We are developing our payment processor to be able to process transactions through NFC. This will allow users to pay with their mobile phones. One user generates transaction and its details, and the other user puts their phone nerby to the first user's phone to complete the transaction.
The main sources of revenue are listed below:

- Transaction fees.
- Risk-free auto exchange fees.

Auto-exchange involves risks associated with unfavorable exchange rate fluctuations. We use statistical models to minimize these risks. Optimization of fees through Machine Learning.

6 Tokenomics

Our token is a utility token designed to reduce fees and provide potential long-term benefits to investors.

Ticket Token Crowdsale

We employ a Ticket mechanism in our crowdsale, which enables investors to receive Ticket Tokens instantly during the crowdfunding process. Once crowdfunding is complete, we will develop the Token, and investors can exchange their Ticket Tokens (KSXT) for Tokens (KSX). This feature allows investors to trade their Ticket Tokens before the actual Token is developed. The sale is 100% public, with no allocation for the team, marketing, or other parties. This structure eliminates the possibility of token dumping by the team or other parties.

Phases

We plan to have a maximum number of phases for the crowdfunding process. This approach provides flexibility for us to initiate phases with specific prices and token availability. Various factors will influence the price and the number of tokens available in subsequent phases. We aim to be adaptable to market conditions. It is possible to conclude crowdfunding even if not all phases are completed, but there will be at least a fixed number of phases we announce. We intend for each phase to last no longer than a fixed period of time and sell no more than a fixed number of Ticket Tokens in each phase.

Phase

Each phase will be divided into buckets (subphases), with each bucket having a designated price and a specific number of available tokens t. Every bucket will be accessible for a fixed period of time, and there will be a number of buckets in each phase. The bucket's price will not exceed the starting price of the phase increased by a fixed percentage x. The tokens in each bucket will be equally distributed among all buckets in the phase. However, if a party wishes to purchase more than specific number of Ticket Tokens, they will not be restricted by the number of tokens available in each bucket, and such a party will be able to reduce the number of buckets remaining in the phase.

Price

- Start: 1KSXT = p_0USDC
- We can gradually increase the price of KSXT in subsequent phases and during the current phase, but we cannot decrease it. Let p_i be the previous price; then, the price of the next phase is $p_{i+1} \in [p_i, (1 + x) \cdot p_i]$.

Total supply

The total supply will be determined by the number of phases and the quantity of tokens sold in each phase, but there will be no more than a fixed supply of tokens available. The total supply may be less if we decide not to execute all phases.

Ticket Token Utilities

- 1KSXT can be exchanged for 1KSX in the future
- From the outset, investors receive an asset that is available for exchange

Token

Fees

A 1% fee will be deducted from the value of every Token transaction. This fee structure favors long-term holders of KSX Token.

- 0.5% of the transaction value is **burned**
- 0.5% of the transaction value is **collected** in the Staking contract for the benefit of KSX Token stakers

Token Utilities

- A deflationary mechanism is in place
- A staking mechanism for long-term holders is in place
- Reduces fees for KSOX services
- Serves as a reputation measure for DAO
- Burns along with KSOX gains

Token will be bought and burnt from the part of collected gains of the project.

7 Security

Thanks to our architecture, at no stage of the order processing do we control the client's funds. Our responsibility is in verifying the signature and updating the contract state guarded by the network consensus. The contract only accepts computation results along with their proof of correctness. The source code of the smart contract and the provable part is open-source. Considering the above, our project is secure and even in the event of unauthorized takeover of our servers, clients' funds are safe because we do not have control over them.

8 Summary

KSOX Exchange is an innovative project that fully leverages the advantages of the state-of-the-art technologies. Considering the overall architecture and market needs, the project aims to deliver value to users who appreciate security and anonymity. Additionally, it aims to simplify the already complex processes of using blockchain networks so that our clients can focus on their matters, rather than integrating with the blockchain. We want to contribute to the initiation of an era of financial applications that do not require trust from clients.