Contents

1 Introduction **3**

2 Methodology **4**

3 Volentix **4**
- 3.1 Venue dynamic community platform . . . 4
- 3.2 Verto wallet . . . 5
- 3.3 Vespucci analytical engine . . . 5
- 3.4 VDex . . . 5
- 3.5 VTX . . . 5

4 Architecture **6**
- 4.0.1 Overview . . . 6
- 4.0.2 Operating system . . . 6
- 4.0.3 Inter Contract Communication . . . 7
- 4.0.4 Side Chains . . . 7
- 4.0.5 Liquidity . . . 8
- 4.0.6 Hashed timelock contracts (Atomic Swaps) . . . 8
- 4.1 Network Topology . . . 8
  - 4.1.1 Nodes . . . 8
  - 4.1.2 Aggregators . . . 9
  - 4.1.3 Latency . . . 9
- 4.2 Order Book . . . 9
  - 4.2.1 Example Data structures . . . 9
  - 4.2.2 On-Chain order book . . . 10
  - 4.2.3 Off-Chain order book . . . 10
  - 4.2.4 Decentalization process of order book settlement . . . 10
- 4.3 Order settlement . . . 10
- 4.4 VTX . . . 11
  - 4.4.1 Token creation . . . 11
  - 4.4.2 Initial Token offering . . . 11
  - 4.4.3 Fee model . . . 12
- 4.5 Inter blockchain communication . . . 13
- 4.6 Security model . . . 13
  - 4.6.1 Introduction . . . 13
  - 4.6.2 General actions to be taken . . . 14
  - 4.6.3 Contract security . . . 14
  - 4.6.4 Malware detection by auditing processes . . . 14
  - 4.6.5 Random diversification . . . 14
  - 4.6.6 Multiple factor identification . . . 15
  - 4.6.7 Logs . . . 15
  - 4.6.8 Transaction as Proof of Stake (TaPoS) . . . 15
  - 4.6.9 Double spend . . . 15
  - 4.6.10 Front running . . . 15

4.6.11 Forged identities 16
4.6.12 Insufficient Balance 16
4.6.13 Timing attack 16
4.6.14 Other EOS.IO security attributes 16
4.7 Inter chain security 16
4.8 Multi blockchain 17
4.9 User experience 17
4.10 True decentralization 17
4.11 System recovery 18
4.12 Scalable and modular architecture 18
4.12.1 Problem decomposition 18
4.12.2 Minimize state space 18

5 Contributions **18**

6 Risk **18**

7 Aknowledgements **19**

8 Conclusion **19**

VDex White Paper v0.1.0

The Volentix Labs Team
info@volentixlabs.com

August 2, 2018

Abstract

VDex is a decentralized exchange with the user and community in mind. Using some of the most recent paradigms and established protocols for security, ease of use and multi asset support, this low friction peer-to-peer exchange abides by open standards and ensures a harmonious and seamless flow among decentralized applications. Built with a collection of smart EOS.IO contracts, it contains easy to use options for security, anonymity, speed of payment, liquidity and profit margin. Open order books support integration with other decentralized exchanges, in effect producing a massive decentralized exchange of exchanges, increasing the liquidity and effectiveness of all the connected exchanges. VDex is also a pillar in the Volentix ecosystem, a network of DApps whose synergy results in even higher liquidity for VDex users.

1 Introduction

VDex is a distributed exchange that provides a highly customizable environment for speed, cost, anonymity, security, and scalability. Power users are enabled

with the freedom to choose and thrive, while new users feel welcomed and free from the risks inherent in a centralized system. The growth of the product resides in a flexible architecture able to adopt the best practices of 2nd generation blockchain applications. The task of building VDex mainly resides in successfully merging and integrating today's best protocols, paradigms and patterns to match the Volentix requirements on top of the 3rd generation blockchain, EOS.IO decentralized operating system.

2 Methodology

All assumptions made in this paper are verified by prototyping with our custom EZEOS software, built with EOS.IO's ***cleos*** command line tools. This software resides at: https://github.com/Volentix/ezeos

3 Volentix

The Volentix ecosystem which consists of DApps which improve the effectiveness of each other. The "four pillars" are an initial set of DApps which support the entire Volentix ecosystem, each in a way specific to their own needs. From the perspective of VDex, the purpose of Volentix DApps is to grow the user base of VDex and thus increase the liquidity available to all users.

- **Venue**
  Grows the Volentix community
- **Verto**
  Enables funds to be continually maintained by the user while using VDex
- **Vespucci**
  Increases trust in the tokens available on VDex
- **VDex**
  Provides liquidity between all crytocurrencies

3.1 Venue dynamic community platform

Venue is a platform which brings together members of the Volentix community and to facilitate distribution of VTX and bring greater awareness to the Volentix Project.

The first iteration of Venue is a platform for distributing VTX. VTX rewards can be earned in exchange for participating in community building campaigns, submitting bug fixes, or claiming bounties. Included in this webapp are leaderboards and live metrics showing user participation. The first signature campaign was launched on the *https://bitcointalk.org/* forum on July 13th, 2018. Visit ***https://venue.volentix.io*** for more information.

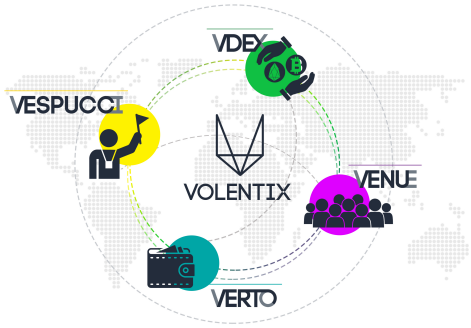

Figure 1:

3.2 Verto wallet

Verto is multi-currency wallet for VDex. It is the central component for all Volentix platforms and is necessary to ensure users continually and safely maintain custody of their private keys. Verto employs a system of smart contracts to maintain the state between two trading clients, the simplest operations being done with atomic swaps.

3.3 Vespucci analytical engine

Vespucci is an application that provides information and analytics about tradeable digital assets, as well as a tool to graph and compare tradeable digital assets. Vespucci increases liquidity by boosting user confidence and bringing more users to the Volentix platform. Vespucci aggregates data presently scattered amongst blockchain explorer sites, chat rooms, and websites, as well as using historic trading records, forum sentiment and plan analysis, trading trends, developer activity, digital asset distribution and governance information, terms and conditions, and historic address balances.

3.4 VDex

VDex provides crypto currency exchange services directly from the user's Verto wallet where both public and private keys are locally managed. The swapping of crypto assets does not involve the temporary custody of the tokens by a central operator, therefore VDex also allows users to trade crypto assets with others without losing custody of the tokens they hold. Furthermore, VDex allows digital tokens to remain on the originator's wallet until all required transactions are completed. VDex supports trading in many cryptocurrencies and hosts multiple liquidity pools to accomodate different types of markets. VDex distinguishes itself with a unique order book decentralization system , multiple liquidity pools and extensive security measures and user awareness mechanisms with regards

to trading in these pools/networks.

3.5 VTX

VTX is an enabling currency that flows through the pillars to enables them to interact.

4 Architecture

4.0.1 Overview

Cross-chain, inter-wallet transactions are done by providing wallets with the contracts or scripts to transact in any cryptocurrency. The transaction between Bob and Alice involves each sending funds toward the other's accounts using these provided contracts best described in the original atomic swap paper.[15] These contracts have guarantees toward each other by the means of shared secrets within timeouts or a refund occurs. The various means of ensuring collateral for a particular transaction or augmenting liquidity of a network will be described later in this document.

4.0.2 Operating system

EOS.IO is an operating system-like framework upon which decentralized applications can be built. The software provides accounts, authentication, databases, asynchronous communication and scheduling across clusters. Components and protocols are already built into the platform, and just a subset can be used to satisfy VDex requirements. VDex initially benefits from the standard features offered by EOS.IO such as account and wallet creation and the recovery of stolen keys, but will subsequently implement the protocols for the creation of decentralized exchanges through its contracts and provided tools [10]

1. **Context Free Actions**
   Most of the scalability techniques proposed by Ethereum (Sharding, Raiden, Plasma, State Channels) become much more efficient, parallelizable, and practical while also ensuring speedy inter-blockchain communication and unlimited scalability. A Context Free Action involves computations that depend only on transaction data, but not upon the blockchain state. ex: Parallel processed signature verifications

2. **Binary/JSON conversion**
   EOS contracts combine the human readability of JSON with the efficiency of binary.

3. **Parallelisation and optimisation**
   Separating authentification from application allows faster transaction times and increases bandwidth. EOS.IO blocks are produced every 500 ms.

4. **Web Assembly(WASM)**
   Web assembly enables high-performing web applications while also secures each applicatios in its own sandboxe. This provides VDex with network access and filesystem namespace restrictions, enforced rule-based execution, and better control over what processes are spawned.

5. **Rust/C++ contracts**
   At the time of this writing, C++ has best tooling for and execution speeds for WASM. C++ is a much more mature language than for instance Solidity used for Ethereum contracts. It also has better debugging support as well as libraries that have been tested over the years and provide reliable functionality. The EOS codebase has also very heavy usage of templates. For example, C++ allows the use of templates and operator overloading to define a runtime cost-free validation of units. Managing memory is much easier building smart contracts because the program reinitializes to clean state at the start of every message. Furthermore, there is rarely a need to implement dynamic memory allocation. The WebAssembly framework automatically rejects any transaction addressing memory wrong. Since EOS.IO contracts use C++14 one can resort to smart pointers if dynamic memory allocation is needed. The first implementation of PARSEC is in Rust.[20]

6. **Schema defined messages and database**
   Service contracts are standardized to guarantee a baseline measure of interoperability associated with the harmonization of data models. The *Standardized Service Contract* design principle advocates that service contracts be based on standardized data models. Analysis is done on the service inventory blueprint to find out the commonly occurring business documents that are exchanged between services. These business documents are then modeled in a standardized manner. The Canonical Schema pattern reduces the need for application of the data model transformation design pattern. [9]

4.0.3 Inter Contract Communication

Data is shared between contracts via an oracle which creates a transaction embedding the data in the chain. "An oracle, in the context of blockchains and smart contracts, is an agent that finds and verifies real-world occurrences and submits this information to a blockchain to be used by smart contracts." [3] Every node has an identical copy of this data, so it can be safely used in a smart contract computation. Oracles push the data onto the blockchain rather than the smart contract pulling the information. Most reading of the data is

done via polling **nodeos** (the blockchain instance) to monitors the blockchain's state and performs certain actions in response.

4.0.4 Side Chains

In EOS.IO, issuing tokens is a process which creates a sidechain. Sidechains are emerging mechanisms that allow tokens and other digital assets from one blockchain to be securely used in a separate blockchain and then be moved back to the original blockchain if needed. There can be multiple side chains where different tasks are distributed accordingly for improving the efficiency of processing. For inter-blockchain communication, the EOS.IO protocol creates a TCP like communication channel between chains to evaluate proofs. For each shard (a unit of parallelizable execution in a cycle), a balanced merkle tree is constructed of these action commitments to generate a temporary shared merkle root; this is done for speed of parallel computation. The block header contains the root of a balanced merkle tree whose leaves are the roots of these individual shard merkle trees. [10]

4.0.5 Liquidity

In order for a token to effectively partake in the global token economy, its trading volume must cross a critical barrier where the matches between buyers and sellers become frequent enough to ensure a stable pool of "coincidences of wants" [11]. This reliability within an exchange is known as liquidity. We say a token is liquid if it is easily possible to buy or sell it without considerably affecting its price.

1. Implementation of the Loopring protocol with the use of EOS.IO contracts acting as nodes.[25]
2. Implementation of the Bancor algorithm used to bring stability to the token.[11]
3. Toggles between these protocols and atomic swaps(HTLC) according to the Vespucci analysis on the VDex network.

4.0.6 Hashed timelock contracts (Atomic Swaps)

A Hashed Timelock Contract (HTLC)[15] is a type of smart contract enabling the implementation of time-bound transactions. Users are offered a variable lock-in period for their transations, with a discount on the transaction fee in exchange for choosing a slightly greater lock-in period.

4.1 Network Topology

4.1.1 Nodes

Nodes are the endpoints of the VDex network. Their function is to:

1. Act as a portal to VDex trough the Verto wallet.
2. Merge order book information with others.
3. Settle order book.
4. Manage order cancellation.
5. Assign timeouts for the **Raft** Protocol.
6. Initiate contract for orders that have been filled.

Nodes earn a portion of the fee for each transaction. If a user has sufficient funds and posessing a good track record, their Verto wallet can act as a node.

4.1.2 Aggregators

The VDex aggregators are dedicated Volentix servers for simulator and security purposes. One of their functions is to pull logs and order book data from nodes into sparse distributed representations for hierarchical temporal memory as intrusion [16] analysis for detecting anomalies in the system such. The aggregators also are host to other components such as metachain ledgers[4] blockchain scrapers and Vespucci components.

4.1.3 Latency

EOS.IO has low latency block confirmation (0.5 seconds).[10] This latency can be provided if the currencies being traded are issued from blockchains that are equally fast, otherwise the transaction is as fast as the slowest block chain (for example a bitcoin block takes 9 minutes to mine at time of this writing). On completion of the transaction, a transaction receipt is generated. Receiving a transaction hash does not mean that the transaction has been confirmed; it means only that a node accepted it without error, although there is also a high probability other producers will accept it.

4.2 Order Book

An order book is a list of orders that VDex uses to record the interest of buyers and sellers. Our matching engine uses this book to determine which orders can be fulfilled. Order books can be tailored according to cost, security and speed and according to which protocol is used to settle the book to cater to these requirements. For instance the Loopring protocol already allows for its orderbook to be modified in order to work with other orderbooks.[25] The same can be done for other non restrictive but efficient protocols such as PARSEC, RAFT or Bancor which each offer their own very distinct advantages. Loopring for instance has very good front-running protection steps by checking for subrings in its ringbuffer(FIFO) data structure, while the Bancor formula can be used to ensure coin stability or create liquidity. On the other hand, PARSEC or RAFT offer the simplest and most efficient decentralization solutions research

has yet to offer. The ability to modify order book settlement methods on the fly according to rule based on model built by oracles residing on aggregators provides another precautionary measure to ensure the homeostatis of VDex.

With this in mind, containers provided by EOS.IO provide best performance.[17] and using these containers to unify the order book morphology is optimal.

4.2.1 Example Data structures

1. Loopring FIFO First-in first-out circular buffer, as suggested by the protocol. Nodes can design their order books in any number of ways to display and match a user's order. Follows an OTC model, where limit orders are positioned based on price alone. [25]

2. EOS.IO persitence API The order book takes advantage of the powerful multi-index container shared among nodes through the same EOS.IO account.

4.2.2 On-Chain order book

An on-chain order book is a record of offers residing on the wallet (node) chosen to settle the order book. It resides in a persistent database on each node subscribing to the same account as all the other nodes(wallets).

4.2.3 Off-Chain order book

Residing on the aggregator, offline order books serve for simulator and security purposes. Built within the period between two order book settlements, this books is gathering logs from as many nodes as possible building its ownversionof the orderbook. At the moment a node settles the on-chain order book, both order books should be identical.

4.2.4 Decentalization process of order book settlement

To ensure decentralization, nodes take turns to settle the Order Book. The settling node must be designated by the protocol and all order book entries from all nodes must be available to the settling nodes. The RAFT[7] and PARSEC[20] protocols offer elegant and simple solutions. The concepts of RAFT are easy to implement and have been around since the time of PAXOS[16], while PARSEC is fairly recent but more efficient, using directed acyclic graphs, eliminating the need for copying logs.

4.3 Order settlement

In the case of a FIFO container for the order book, settling occurs as such:

1. Check for subrings this operation promotes security by checking the order of the ring is the same as the initial one.

2. Load Balancing (Fill rate/stock availability)
3. Rate is equal or less than the original buy rate set by user (Definition of limit orders)
4. Process cancellations
5. Orders are scaled according to the history of filled and cancelled amounts

4.4 VTX

VTX is the cryptocurrency used on the vDex exchange. It can be used within all of the four pillars of the Volentix system:

1. To pay transaction fees on the VDex.
2. To vote on proposals submitted to the network, using the voting rights allocated to VTX holders.
3. To submit proposals to the network.
4. To stake support for project building, liquidity, or proposal reviews.
5. As a fee redistribution token on the VDex exchange.
6. To incite users to partake in order book settlement.
7. To reward participants in the consensus process and in Venue campaings.

4.4.1 Token creation

The **eosio.token** contract from the EOS.IO framework is used to issue 2.1 billion EOS.IO compliant tokens with a supply of 1.3 billion.

4.4.2 Initial Token offering

Distribution

1. **The founders**
   The initial promoters of the project. Conceptualization, plan and initialization.
2. **Prior work, core team**
   Consists software engineers, blockchain analysts, legal experts and business development specialists. In addition to the core team, Volentix has a board of advisors consisting of top experts from a variety of fields relevant to the project.

3. **Decentralized treasury**
   Publicly controlled funds for community projects, business operations and salaries for ongoing development. A small percentage or every transaction goes back into the treasury.

4. **Ongoing core development**
   Incentives for core team. These are bonuses given throughout the project after milestones have been reached.

Table 1: Token distribution

Percentage of issued tokens	**price**	**Timelock**
35% Future Decentralized Treasury		
5% Initial funding	0.000016-0.000020 BTC	
28% Distribution	0.000021 BTC+	
12% Founders		yes
10% Prior Work, Team and Advisors		yes
10% Ongoing Core Development enticement		

Justification of token crowdsale vote Pre-sale VTX holders, founders and the core team are given the opportunity to participate in an ***augmented vote***[6] to know if the Volentix community wants a crowdsale or not. Augmented votes require voters to take an easy test which proves they acknowledge facts concerning crowdsales. Participants to the elaboration of this test grid compile facts linked to publications regarding crowdsales and earn VTX in the process. The resulting vote is therefore ***augmented*** by ensuring all voters have the same established knowledge regarding this issue.

Multi signature account The raised funds are to be deposited to a multi signature account owned by the founders that can only be accessed with the consent of at least two of three signees.

4.4.3 Fee model

Transaction fees

1. Fees are to be collected in any currency.

2. Fees can be assigned to specific accounts

3. A small fee collected for general maintenance by developers

Lock-in period fee
A Hashed Timelock Contract (HTLC)[15] is a type of smart contract enabling the implementation of time-bound transactions.

1. Users can lock in funds for 24hrs and have a free transaction

2. Users can lock in funds for 24hrs+ and generate VTX

Order book settlement fees A wallet can become a node and earn VTX through order book settlement.

EOS.IO The following considerations are applicable for deploying the exchange on the EOS.IO platform

1. Deploying a contract has a cost but is free to use.
2. Developers have to stake EOS.IO tokens to deploy contract. After the contract is destroyed, the locked tokens are returned.
3. DApps must allocate resources to their contracts, memory, cpu, bandwidth.
4. The choice of who pays the resources is up to the DApp.
5. Multiple messages in one transaction and multiple accounts can be asigned to the same thread.

Budget forecast Since the acceptance of decentralized exchanges has not yet arrived, but is deemed to happen, it is logical to say that the value of VTX increases with demand over a longer period of time, around 5 years.

4.5 Inter blockchain communication

EOS.IO is designed to make Inter-Blockchain-Communication proofs lightweight. For chains with an insufficient capacity for processing the IBC proofs and establishing validity, there is the option to degrade to trusted oracles/escrows. To directly control other currency transactions with an EOS.IO based smart contract, a trusted mutisig wallet holding the currency in escrow is used to persuade the signing/publishing of the currency transaction based on IBC proofs from the originating chain.

4.6 Security model

4.6.1 Introduction

The assumptions made in this section are made from the analysis on gathered information. Full security testing begins after the ongoing prototyping phase.

The security concerns with the VDex environment can generally be categorized as variants of the following scenarios:

1. The attacker executes malicious code within a transaction
2. The order of transactions is manipulated
3. The timestamp of a block is manipulated

4.6.2 General actions to be taken

1. Filter incoming data
   (a) Data from blockchains and order book resolution.
   (b) Data received by nodes
   (c) Data received by wallet
   (d) Data received by aggregators
2. Filter outgoing data
   (a) Cancel order
   (b) Commit

4.6.3 Contract security

1. Keep vast majority of funds in a time-delayed, multi-sig controlled account
2. Use multi-sig on the hot wallet with several independent processes/servers double-checking all withdrawals
3. Deploy a custom contract that only allows withdrawals to KYC'd accounts and require multi-sig to white-list accounts
4. Deploy a custom contract that only accepts deposits of known tokens from KYC'd accounts
5. Deploy a custom contract that enforces a mandatory 24-hour waiting period for all withdrawals
6. Utilize contracts with hardware wallets for all signing, even automated withdrawals
7. Seamless sytem to upgrade broken contracts
8. Ability to pause the functionality of a contract
9. Ability to delay an action of a contract

4.6.4 Malware detection by auditing processes

The system provides insights on rogue processes during the transaction period with AI analysis residing on the aggregators.

4.6.5 Random diversification

By using the RAFT protocol in the election process, a certain level of randomization is aquired with varying length of timeouts. The toggling of protocols is a level of complexity

4.6.6 Multiple factor identification

As in many existing applications, this measure is efficient and already known to the public at large.

4.6.7 Logs

Ensure inspection of logs as control.

1. Raft.
2. Anomaly detection with AI(Numenta).
3. Script investigations of certain non token purchases related addresses.

4.6.8 Transaction as Proof of Stake (TaPoS)

1. Prevents a replay of a transaction on forks that do not include the referenced block
2. Signals the network that a particular user and their stake are on a specific fork.

4.6.9 Double spend

A double spend is an attack where a given set of coins is spent in more than one transaction.

1. Send two conflicting transactions in rapid succession into the network. This is called a race attack.
2. Pre-mine one transaction into a block and spend the same tokens before releasing the block to invalidate that transaction. This is called a *Finney* attack.
3. Own 51+% of the total computing power of the network to reverse any transaction, as well as have total control of which transactions appear in blocks. This is called a 51% attack. This is impossible according to EOS.IO, Loopring or Raft. If a block producer takes an unreasonable amount of runtime or is not profitable enough, the process is blacklisted.[25]

4.6.10 Front running

To prevent someone from copying another node's trade solution and have it mined before the next supposed transaction in the pool, a higher fee per transaction is charged.
The major scheme of front-running in any protocol for order-matching is order-filch: when a front-runner steals one or more orders from a pending order book settlement transaction. EOS.IO and loopring both have remedies to this. In both cases keys are not part of the on-chain transaction and thus remain unknown to parties other than the order book settling node. Nodes dictate how they manage orders; each node uses a different solution for resolving the order books, inducing another level of randomness to promote security.

4.6.11 Forged identities

Malicious users acting as themselves or forged identities could send a large number of small orders to attack Loopring nodes. However, most of these orders are rejected for not yielding satisfying profit when matched. Again, nodes should dictate how they manage orders.

4.6.12 Insufficient Balance

Malicious users could sign and spread orders whose value is non-zero but whose address actually has zero balance. Nodes could monitor and notice that some orders' actual balance is zero, update these order states accordingly and then discard them. Nodes must spend time to update the status of an order, but can also choose to minimize the effort by, for example, blacklisting addresses and dropping related orders.

4.6.13 Timing attack

Timing attacks are a class of cryptographic attacks through which a third-party observer can deduce the content of encrypted data by recording and analyzing the time taken to execute cryptographic algorithms. The randomness of time-outs in the raft algorithm prevents this, for instance, at the orderbook level. Concerning the wallet, the careful use of elliptic curve cryptography can prevent backdoor intrusion.

4.6.14 Other EOS.IO security attributes

1. No uses of mutex or locks for on-chain parallellisation
2. All accounts must only read and write in their own private database

4.7 Inter chain security

Transactions sent to a foreign chain require some facilities on the foreign chain to be trustless. In the case of two EOS.IO based chains, the foreign blockchain

runs a smart contract which accepts block headers and incoming transactions from untrusted sources and is able to establish trust in the incoming transactions if they are provably from the originating chain. For chains with an insufficient capacity for processing the IBC proofs and establishing validity, the options degrade to trusted oracles/escrows.

4.8 Multi blockchain

Multi-blockchain information can be obtained by aggregating blockchain timelines in parallel order (with variance in the frequency of when the state is changed) into a comprehensible data structure. This enables a system to trigger multichain load balancers, transfer states, draw data outputs from smart contracts, and trigger execution of transactions on foreign blockchains. Relative block distance, relative global state, and timestamped events are recorded on a global ledger to optimize and confirm transactions before they actually happen on the native chain. This could be used to determine block production coincidence between chains to choose optimal liquidity.[4]

4.9 User experience

Simulator In an effort to provide a better and safer user experience, a VDex trading simulator is provided with convenient scenarios.

Templates Easy to use templates for standard transactions is also provided.

Comprehensive customizeable and detailed interface

1. Shows the entire market and fluctuations
2. Shows wallet: balance and previous transactions.
3. Shows detailed history with built in tax calculator.
4. Contains toggles for advanced features.

4.10 True decentralization

Decentralization is the process by which planning and decision-making are distributed or delegated away from a central, authoritative location or group.

EOS.IO is a free, open source, scalable infrastructure for decentralized applications. It tries to ensure a fair and transparent block producer (BP) election process utilizing a democratic delegated proof of stake (DPoS) consensus. A pragmatic understanding of the complexity of the task at hand combined with the unpredictability of human interactions implies that a small amount of centralization may be required in the initial stages. The criteria of decreasing the *incidences of centralization* with time is necessary. Regardless, the exchange

itself possesses several mechanisms to promote decentralization. For instance the system for electing nodes when solving order books does not use a shared central clock or (DPoS) but is based either on random timeouts for the determination of leaders in an election (RAFT), or using graph theory (DAG) in the case of the PARSEC protocol.

4.11 System recovery

By ensuring that the latest version of the ledger is always put forward, the RAFT and PARSEC protocols provide a robust system for recovery in the case of node failure. Security measures are also provided in case of IBC for trading with native blockchains. In the case of non-identification of the chain, a fall back occurs to the native rules to wait until the next block or default to a short time lock.

4.12 Scalable and modular architecture

To secure the potential for innovation, the principles, concepts and paradigms proposed by components of the system must favour decoupling of technologies. Since creating and maintaining distibuted and decentralized systems is very complex we must use different strategies:

4.12.1 Problem decomposition

Problem solving strategy of breaking a problem up into a set of subproblems, solving each of the subproblems, and then composing a solution to the original problem from the subproblem solutions.

4.12.2 Minimize state space

Dynamic programming and templating are hard because of complexity and debugging challenges. Nesting conditions can also seem unimportant for normal program execution. Special attention to these patterns and details in the initial design will maximize efficiency while allowing for easy replacement or addition of components. In the context where CPU, bandwidth and RAM are monetized, this is even more important.

5 Contributions

The public software repository for Volentix is https://github.com/Volentix. All contributions and suggestions to these repositories will be reviewed and integrated.

6 Risk

The sheer number of transactions VDex will eventually process is hard to visualize, but in the context of the growing interest for decentralized exchanges, more transactions equate with more risk. Managing the risks of handling currency, initially and still today, proves to be a challenge to centralized providers. It is long and arduous process to aquire the trust of a network.
The evolution of decentralized exchanges should not be expected to be any different, but unlike centralized exchanges, dexes can have the support of the open source developer community which quickly contributes forward solutions to resolving problems and enhancing the exchange.

7 Aknowledgements

Thanks to professor Yiannis Emiris from the Department of Informatics and Telecommunications at the National and Kapodistrian University of Athens for comments on this document.

8 Conclusion

Constructs, concepts and protocols that stand out by their simplicity while retaining their effectiveness helps implement a modular design which enhances the system with the capacity to easily add or replace components with the prospects of carefully advancing functionality as micro services are created. Although certain assumptions made in this paper still remain to be verified, a very distinctive direction for VDex architecture has been distilled as a highly flexible and modular MVP capable of adaptation and reaction to the changing technological ecosystem. For this, provisions of the EOS.IO operating system, Loopring, Bancor, RAFT and PARSEC protocols have been retained.

Bibliography

References

[1] Aelf, *A multi-chain parallel computing blockchain framework*, (2018).

[2] ARK, *A platform for consumer adoption*, (2018).

[3] blockchainhub.net, *blockchain-oracles*, (2017).

[4] BlockColliderTeam, *Block collider white paper*, (2018).

[5] V. Buterin, *Ethereum: a next generation smart contract and decentralized application platform*, (2013).

[6] S. Cormier, *A machine based societal model for curbing citizen cynicism*, (2017).

[7] J. O. Diego Ongaro, *In search of an understandable consensus algorithm*, (2018).

[8] M. Duncan, Quale, *Halo platform*, (2018).

[9] T. Earl, *Soa principles of service design*, (2016).

[10] EOS.IO, *Eos.io technical white paper v2*, (2018).

[11] G. B. Eyal Hertzog, Guy Benartzi, *Bancor protocol: Continuous liquidity for cryptographic tokens through their smart contracts*, (2018).

[12] S. D. K. M. T. S. H. Garcia-Molina, *The eigentrust algorithm for reputation management in p2p networks*, (2018).

[13] M. R. Garrick Hileman, *Global cryptocurrency benchmarking study*, (2017).

[14] Komodo, *An advanced blockchain technology, focused on freedom*, (2018).

[15] K. Kurokawa, *Atomic cross chain transfer, an overview*, (2015).

[16] L. Lamport, *The part time parliament*, (1998).

[17] D. Larimer, *eosio.boot telegram chat*, (2018).

[18] Q. Liquid, *Providing liquidity to the non-liquid crypto economy*, (2018).

[19] S. R. M.P.M-S, Aniket Kate Matteo Maffei, *Concurrency and privacy with payment-channel networks*, (2017).

[20] F. H. Q. M. S. S. Pierre Chevalier, Bart lomiej KamiÂĂtnski, *Protocol for asynchronous, reliable, secure and efficient consensus (parsec)*, (2018).

[21] SingularityNET, *A decentralized, open market and inter-network for ais*, (2018).

[22] M. M. Timo Hanke and D. Williams, *Dfinity technology overview series consensus system*, (2018).

[23] A. B. Will Warren, *0x: An open protocol for decentralized exchange on the ethereum blockchain*, (2017).

[24] G. Wood, *Ethereum: A secure decentralised generalised transaction ledger.ethereum project yellow paper*, (2014).

[25] F. Zhou, Wang, *Loopring: A decentralized token exchange protocol*, (2018).

[9] [3] [10] [5] [24] [13] [25] [23] [19] [11] [12] [14] [2] [8] [18] [21] [22] [7] [1] [4] [6]