Republic Protocol

A decentralized dark pool exchange providing atomic swaps for Ethereum-based assets and Bitcoin.

December 18, 2017

Taiyang Zhang, Loong Wang

Abstract

The market capitalization and trading volume of cryptocurrencies is growing rapidly every month. With institutional investors arriving into the cryptocurrency market, the development of alternative trading systems is critical for trading large blocks of cryptographic assets while maintaining minimal price slippage and market impact.

We introduce Republic, a decentralized open-source dark pool protocol facilitating atomic swaps between cryptocurrency pairs across the Bitcoin and Ethereum blockchains. Trades are placed on a hidden order book and are matched through an engine built on a multi-party computation protocol. This provides order execution without exposing market sensitive information such as price and volume at a certain position, which would provide an advantage to other traders.

Republic removes the need for a trusted intermediary to operate a dark pool and provides crypto-economic incentives through a protocol token for governance; enabling the development of a secure, decentralized, scalable dark pool protocol capable of handling billions in trading volume daily.

Introduction

The advent of blockchain technologies has enabled the development of an entirely new class of assets backed by cryptographic verification. Bitcoin (BTC) and Ethereum (ETH) are two blockchain-based cryptocurrencies which, as of eclipse the aggregate market capitalization of all other cryptocurrencies.

In November 2017, the volumes for BTC and ETH trades exceeded USD $181B (not including over-the-counter and trades executed on private forums). This statistic, coupled with the announcements of Bitcoin futures markets from CME Group and NASDAQ, signals interest from institutional investors looking to gain exposure to digital cryptographic assets. With institutions and HNWIs looking to deploy vast amounts of wealth into cryptocurrencies, we must develop the underlying infrastructure to support such volumes.

At a fundamental level, dark pools are private exchanges where financial assets and instruments are traded and matched by an engine running on a hidden order book. These exchanges are primarily created to serve institutional or HNW retail investors who require a system where significant volumes of assets can be block traded with minimal price slippage. Dark pools are estimated to represent approximately 15% of all trading volume of all US stock trades [6]. Extrapolating this statistic for BTC and ETH volumes, a dark pool for such has the potential to execute USD $27.2B of orders monthly.

We introduce the Republic Protocol which facilitates the exchange of Ethereum, ERC20 and Bitcoin cryptocurrencies through a decentralized dark pool. This is enabled through research within subfields of cryptography such as secure multi-party computation, which allow us to develop a matching engine to run on the distributed hidden order book. We facilitate cross-chain trades through atomic swaps and implement proper economic incentives to ensure these trades are executed thoroughly. Compared to a centralized dark pool or exchange, the Republic Protocol removes the risk of asset theft, confiscation or possibility of interference from a malicious exchange operator. This leads to greater trust between institutional investors placing block orders and dark pool exchanges leveraging the Republic protocol. Additionally, the Republic Protocol is available universally and is highly transparent with regards to how the underlying protocol operates.

Elementary Components

- Decentralized hidden order book
    - A decentralized, hidden order book.
- Decentralized order matching
    - Matching orders without knowing the underlying details
- Atomic swap infrastructure
    - The ability to swap between Bitcoin, Ethereum and Ethereum-based tokens without trust.
- Protocol token
    - The REN token

Motivation

- Infrastructure for block orders
- Cross-chain trades
- Trustless, equitable access to dark pools
- Centralization risk

Republic Protocol

How the Republic Protocol works

The primary technical goal of the Republic Protocol is to enable a decentralized network of nodes to match *orders*, without knowing anything about the orders. While it might seem like this is impossible, it can be achieved by applying cryptographic techniques that have been thoroughly researched over that last 30 years; modifying them to be suitable for the world of decentralized computation.

The Republic Protocol uses the *Shamir Secret Sharing Scheme* [1] to break down orders into a large number of *order fragments*, and distributes them throughout the network. Orders cannot be reconstructed unless a majority of the order fragments are recombined. To prevent this from happening, the Republic Protocol defines an Ethereum smart contract called the *Registrar* that organizes nodes into a network topology that makes it unreasonably difficult for an adversary to acquire the enough of the order fragments to reconstruct an order. As long as traders respect the network topology defined by the Registrar, their orders will be safe. If they fail to do so, only their own orders are at risk of exposure.

Using order fragments from two different orders, a node can cooperate with other nodes that hold other order fragments for the same two orders to perform a decentralized computation that will determine if the two orders match. The decentralized computation does not expose the order fragments, and performs a random scaling of the final output [2][3]. This prevents nodes from reconstructing the original orders, and prevents them from using the output to infer anything about the orders. A *Zero knowledge proof* is used to verify the integrity of the computation, without revealing any information. These proofs are simple and efficient, allowing them to be performed by an Ethereum smart contract called the *Judge* [3].

After two orders have been matched, an atomic swap is initiated between the two traders over the Republic Swarm Network, a decentralized peer-to-peer network. Using standard asymmetric encryption primitives, the details of the atomic swap are kept secure.

System Properties

The Republic Protocol provides the following properties:

1. The identity of the traders is secure within the Republic Dark Pool. The underlying cryptocurrency that is being traded may provide different limitations for privacy.

2. Traders do not have to remain connected to the network while their orders are being matched. Once an order is placed, nodes will run the matching computation until a match is found, or the order is expired (either manually, or by passing a deadline designated by the trader).

3. An order is secure until it is matched. After being matched, some details of the order are revealed to the matching parties. This is the natural limit of security for an order, since both parties know what they submitted, and both parties need to know when a match has occurred. Note that information disclosed in these cases does not provide any informational advantage to either party.

4. The total liquidity of the Republic Dark Pool cannot be reasonably estimated by any participant.

Assumptions

The Republic Protocol is built on the following assumptions:

I. There exists a trusted third-party that will always perform computations honestly, but has limited computational power (i.e. Ethereum).

II. Participants act rationally and will not participate if there is no financial incentive to do so, and will attempt to maximize their own profit. In this way, we do not assume that a participant will act honestly if they can maximize their profit by acting maliciously.

Adversarial Assumptions

The Republic Protocol makes the following adversarial assumptions:

I. Adversaries cannot corrupt the trusted third-party defined previously by Assumption (II). Concretely, an adversary cannot subvert the correctness of computations done by the Ethereum network. All platforms built on Ethereum need to make this adversarial assumption.

II. Adversaries have limited financial, and computational, powers. Limited financial powers are a reasonable assumption to make in the real world, and computational powers are naturally limited by financial powers.

III. Computationally hard problems used to construct cryptographic primitives are sufficiently secure. This assumption is made by all blockchains that utilize any form of cryptography, including Bitcoin and Ethereum.

Security Model

Defining a security model allows us to analyze the security guarantees provided by the Republic Protocol. The Republic Protocol makes use of the *real* vs. *ideal* paradigm; analyzing the security of a *real* world decentralized protocol with respect to some non-existent *ideal* world in which there is a trusted, and incorruptible, third-party that can be used to handle all sensitive information and perform all sensitive computations (this is not the same as Ethereum, since all transactions and data on Ethereum is publicly available). The security of the Republic Protocol can be demonstrated by showing that any possible attack in the *real* world is also possible in the *ideal* world. Since the *ideal* world is trivial to define, the *real* protocol is secure by implication. This approach to security analysis is typical for decentralized computation protocols in which there are active and passive adversaries.

The *ideal* Republic Protocol contains a trusted, and incorruptible, third-party T. Traders submit their orders to T, and T guarantees to never reveal the details of these orders. T constantly attempts to match orders that have been submitted, and when a match is found T informs the respective traders. The traders each submit their cryptocurrencies to T, and if they both do so, T swaps the cryptocurrencies and gives them back to the traders. This completes the trade.

The *real* Republic Protocol is considered secure if, and only if, all attacks on the *real* protocol are also possible on the *ideal* protocol. From the definition of the *ideal* Republic Protocol it is clear that such an equivalence is sufficient.

The Republic Protocol is able to guarantee that, unless the majority of nodes in the network are active adversaries, it is as secure as the *ideal* world protocol. If 50% of nodes are active adversaries, and they are enjoying the attackers best-case scenario, they are able to reconstruct all orders. However, the Republic Protocol ensures that such a best-case scenario is impossible to achieve in the *real* world. In the typical case, 50% of nodes becoming active adversaries would only allow the adversaries to reconstruct 50% of the orders. A more detailed explanation is given in “Attacks and Defenses”.

Decentralized Order Matching

Order matching is the process through which nodes match *orders* against each other without being able to observe the details of the order. To achieve this, traders first breakup their order into a set of *order fragments*. Note that these fragments do not individually represent a fraction of the order's value, they simply represent the separation of sensitive data regarding the underlying order. On its own an order fragment reveals nothing about the underlying order, but when at least half of the order fragments for an order are combined, the order can be reconstructed (see “Attacks and Defenses” for details about protecting against this). Each node performs an order matching computation on order fragments from multiple different orders and combines the results with the results from nodes (who are using different fragments). The fragments are constructed in such a way that, after the computations are applied, the resulting fragments can be combined to reveal, not the underlying orders, but the result of the order matching computations for the underlying orders.

This has several nice properties. For one, only half of the order fragments are needed to reconstruct an order. Nodes are incentivized to avoid collusion (and adversaries have a difficult time subverting this system, see “Attacks and Defenses”). This means that if half of the nodes accidentally die, or leave the network halfway through an order matching computation, the network can still finish the computation. This makes it highly resilient to DDoS attacks, and expected failures.

Order fragments are constructed in such a way that the order matching computations can use any function, applied over a polynomial, and can be involve two or more underlying orders. This allows for very flexible order matching computations. Nodes can match orders based on exact price points, partially match orders (when only some of an order can be matched due to the currently available liquidity), match triplets (or more) of orders to increase liquidity (e.g. the triplet BTC-to-ETH and ETH-to-REN and REN-to-BTC, where no match can be found with only pairs). Assuming the existence of a decentralized, consensus-based, data stream for National Best Bid and Offer (NBBO) data, the order matching computations can even involve orders without an explicit price point.

Winning and Losing

Nodes race to discover order matches. Any match that is found must be registered so that other nodes can see which orders have been closed. The associated traders are notified, and none of the matched orders can be involved in future matches. This is done on the Ethereum network, under Assumption (1). If two orders do not match, they continue to be used in future matching games. If an order cannot be matched before it expires, the associated fee is refunded.

The nodes that combine their outputs to register a match are rewarded a fee, to incentivize their honest participation in the order matching game (see “Incentive Layer”). This also incentivizes them to match as many orders as quickly as possible, since this correlates to a higher reward over time.

The Republic Protocol also includes an Atomic Swapping protocol that is initiated between traders that have had their orders matched. Nodes facilitate passing messages (and where possible, setting up a direct P2P connection between traders) that executes the order. Note that traders cannot be bound to execute on the orders, due to the limited way in which blockchains can communicate (see “Attacks and Defenses” for information about placing false orders). However, using trader bonds, traders can be heavily incentivized to faithfully execute orders.

At no point during order matching, or even after orders have matched, are Republic Protocol nodes capable of revealing the details of an order. Even if a malicious adversary is capable of performing a 51%

attack, the order fragments are distributed in such a way that the adversary is only able to reconstruct 50% of the orders (the higher attack percentage they achieve, the higher the rate of order reconstruction).

REN Tokens

Under Assumption (II), computational nodes must be incentivized to perform the order matching computations. It is unlikely that participants will be willing to run order matching nodes if they have no financial incentive to do so, especially when running and maintaining order matching nodes is not free.

The REN token is introduced to provide this incentivization. It is also used to pay bonds to the Registrar, allowing traders and order matching nodes to participate in the Republic Dark Pool.

Order Fees

Fees provide a decentralized mechanism for the users of the system (i.e. traders) to remunerate those that are providing the computational power (i.e. nodes) necessary to fulfill the needs of the users. This is necessary under Assumption (II).

Traders pay an order fee, in REN, when submitting an order. If the order expires before it is matched, the order fee is refunded to the trader. Any node that participates in the decentralized computation for an order that has been matched receives a share of the order fee (the shares are calculated by evenly splitting the order fee amongst all of the participating nodes).

The order fee is variable, and under Assumption (II), orders with higher order fees will be favored by the order matching nodes. However, nodes have no incentive to ignore an order, especially since they do not know the identity of the trader, nor the details of the order. The only information available to the node is the amount of REN that they will receive for successfully matching this order. Note, all order matches will actually result in two payouts to each participating node, one from each side of the match.

Bonds

Orders are secured by breaking them down into several order fragments that are distributed throughout the network. An adversary attempting to reconstruct orders could join the network with a large number of nodes in the hope that they will receive the majority of the order fragments (we will see later that this is not actually feasible). Similarly, an adversary could submit a large number of false orders (that they do not intend to execute on) in an attempt to probe the legitimate orders.

To prevent this class of Sybil attacks, and provide a simple identity mechanism, traders and nodes must submit a bond in REN before they are allowed to access the network. This bond is associated with a single identity in the Registrar smart contract and the registration status can be queried by anyone. The bond is refunded in full when the trader or node leaves the network. Traders are free to submit a flexible bond, with higher amounts allowing a higher number of parallel open orders (the larger the financial bond, the harder it is to perform a Sybil attack, and so more orders can be submitted safely).

Nodes must submit a bond in REN higher than some globally defined threshold (this threshold can be set as needed, to keep the bond requirement above a sufficiently large financial commitment). Since this threshold is dynamic, nodes can alter the bond amount however they choose but will not be able to participate unless their bond is above the threshold.

During verification, the Challenger and the Provers (usually the trader and a group of order matching nodes, respectively) put their bond on the line. If the Challenger is correct, the Prover that is unable to provide evidence of a truthful computation loses their bond. Likewise, if all Provers are correct, the Challenger loses their bond. In this way, the REN bond also acts as a disincentive to attempt to cheat.

Attacks and Defenses

Order Reconstruction

The security of an order maintained as long as *n/2* of its n order fragments are not discovered by an adversary. If an adversary does acquire *n/2* (or more) order fragments, the original order can be reconstructed. As such, it is important to understand the defenses in place against such an attack.

Nodes in the Republic Dark Pool are partitioned into n disjoint sets, where each order share is randomly distributed to *at most* two nodes in any one set. To model an attack on this topology, we assume that the adversary has full control over which nodes to corrupt (the Republic Protocol enforces that nodes are actually randomly distributed amongst the disjoint sets, meaning that this assumption provides the adversary with more power than they have in reality).

The ideal attack scenario would be where an adversary corrupts all of the nodes in *n/2* sets, guaranteeing that *n/2* order fragments will be acquired for every single order. Assuming an approximately uniform size of each pool, the adversary must control 50% of the network. Note that it is impossible for an adversary to control in which set their nodes will be registered, making this type of attack impossible.

Realistically, when controlling 50% of the network, the adversary is most likely to control 50% of the nodes in all of the n disjoint sets. At this level of control, an adversary has a 0.5 probability of successfully acquiring each order fragment but must successfully acquire *n/2* order fragments to know the order. We can model this as a binomial distribution.

Let X be the number of successfully acquired order fragments, p be the probability of acquiring any one order fragment, and n be the number of attempts that the adversary has for any one order fragment.

$$X \sim B(n,\ 0.5)$$

Because X is binomially distributed with a 0.5 probability of success. It follows that,

$$Pr(X \geq n/2) = 1 - Pr(X < n/2)$$

$$= 1 - \sum_{i=0}^{n/2} C_k^n p^i (1-p)^{n-i}$$

This formulation relies on n, the number of disjoint sets, which is directly proportional to the number of nodes in the Republic Dark Pool.

$$\lim_{n \to \infty} 1 - \sum_{i=0}^{n/2} C_k^n p^i (1-p)^{n-i}$$

$$= 0.5$$

As the number of nodes in the Republic Dark Pool grows, the probability that an adversary is able to reconstruct a single order approaches 0.5. This implies that an adversary that somehow manages to corrupt 50% of the network only manages to discover 50% of the orders.

False Orders

When two orders are matched, both of the matching parties learn that there exists some corresponding order in the Republic Dark Pool (otherwise a match would not have occurred). An adversary can take advantage of this in an attempt to gain insight into the liquidity of the Republic Dark Pool.

Assume that there are n legitimate orders in the dark pool when there is no adversary. To simplify the analysis we also assume, in the favor of the adversary, that the adversary knows the maximum price point of orders in the dark pool (realistically, this is impossible and the adversary would have to make several guesses).

If we assume that none of the legitimate orders have matches, the adversary needs to submit n false orders (at the maximum price point) to discover all orders. Compared to the fees paid by the rest of the network, the adversary needs to match 100% of the financial commitments to order fees made by the network. By Assumption (II) this is not realistic, and becomes more and more difficult as the Republic dark pool is used.

Now we assume that each of the n legitimate orders has exactly one legitimate match, and an attacker has a way of distributing their order fragments in such a way that their false orders are instead matched with a $p{=}50\%$ probability. Again, this assumption is in favor of the adversary, since they cannot actually know how to perform such a distribution.

For a binomial distribution with corresponding probability of success p, the probability of exactly k successes given n trials is given as

$$\frac{n!}{k!(n-k)!}p^k(1-p)^{n-k}$$

For example, if $n{=}100$ and $p{=}0.5$, then the probability is approximately 54%. This shows that only with a substantial commitment to order fees compared to the network as a whole, along with many favorable assumptions, is an adversary able to gain insight into the liquidity of the dark pool.

This analysis does not take into account that there is a limited number of orders that can be submitted by any one trader. To submit a large quantity of false orders a trader would also need to stake a large amount of financial power into bond registrations.

Future versions will also discuss methods by which traders must forfeit their bond if they do not execute on false orders. Taking these three parts of the analysis into account: the high amount of order fees required to gain insight into the dark pool, the high amount of bond required to submit that many orders, and the high amount of bond sacrificed when false orders are not executed, Adversarial Assumption (II) prevents adversaries from being able to expose the liquidity of the dark pool by submitting false orders.

Sybil Attacks

In the Republic Protocol, defending against order reconstruction attacks (and false order attacks) requires associating an identity with a node (or trader). This opens the possibility for an adversary to forge multiple identities, known as a Sybil attack, in an attempt to subvert the network.

To protect against this, all nodes and traders are required to commit a bond in order to register an identity. Under the Adversarial Assumption (2), adversaries have limited financial power, we can be sure that an adversary cannot forge a large number of identities.

For malicious nodes, the bond needs to discourage the registration of a large number of nodes and the acquisition of a sufficiently large number of order shares during the distribution of order shares (see "Order Reconstruction"). For this method to be effective, the bond must be high enough that an adversary cannot register a large number of nodes, but small enough that honest nodes are still able to participate. The bond amount should be globally consistent (all nodes must meet the same threshold) but dynamic, to account for fluctuations in the value of the bonded currency.

For malicious traders, the bond can be used to further discourage the submission of a large number of false orders (see "False Orders"). This is done by requiring that a trader submit orders that point to their registered bond. There is a linear relationship between the bond amount, and the maximum number of orders. Therefore, a trader that submits a bond of B and is allowed M open orders could instead submit a bond of $B/2$ and be allowed $M/2$ open orders.

The registration of bonds will be handled by the Ethereum network, and are incorruptible by Assumption (1).

Republic Terminal

We introduce a web-based decentralized application (DApp) for traders to interface with the Republic Protocol. This real-time terminal provides traders with the capability to place, cancel or amend orders. Users can also view the status and history of their orders, visible only to themselves.

Roadmap

Q1 2018

- P2P overlay network (for network-wide communication)
- REN ERC20 contract
- Atomic Swapping contracts and scripts (support for BTC, ETH, ERC20, ERC721)
- P2P pooling network (for intra-pool communication)
- Registrar contracts (for registering nodes and traders)

Q2 2018: Private + Public Testnet

- Order booking contracts (for opening and closing orders)
- Secret sharing and computation
- Official nodes maintained by Republic
- Trader terminals (software allowing traders to open / expire orders and perform P2P atomic swapping)
- Verifiable secret sharing

Q3 2018: Mainnet

- Full specification for Republic Protocol
- Open-source reference implementation
- Package reference implementation for Ubuntu and Docker
- Partner nodes maintained by Republic partners
- REN powered DAO

Q4 2018 & Onwards

- Support for more chains including Litecoin.
- Implementation of scaling solutions for atomic swaps including Lighting & Raiden.
- Support for High Frequency Trading
- Support for institutions

Summary

The Republic Protocol is able to provide a decentralized dark pool, allowing traders to place different types of orders that are matched against others without exposing the details of those orders. This is made possible by the use of Shamir Secret Sharing [1] and a specialized distributed network that allows nodes to perform order matching computations on order fragments, instead of needing to use the orders directly. In this way, orders are not revealed to nodes within the network [2][3]. Computations are verified through zero-knowledge proofs, and malicious nodes that are discovered are punished by forfeiting their registration bond [3]. This allows traders to trust that their orders will be correctly matched. The techniques used have been peer-reviewed and have had years of security analysis and are well accepted in academic cryptography.

The REN token is introduced as an incentivization token. It is used by traders to offer an order fee, from which order matching nodes are rewarded. In this way, nodes are paid for their computational efforts and traders are able to prioritize important orders by issuing larger fees. To further secure the system, the REN token is used as a bond by traders and order matching nodes, providing identities (while protecting against Sybil attacks) that can be used to rate limit orders and punish malicious order matching nodes.

In the presence of a 50% attack on the network, where an adversary controls 50% of the order matching nodes, they are still only capable of revealing 50% of the orders that flow through the system, without the ability to control which 50% are being revealed. An attempt to probe orders by submitting false orders is also an impractical attack surface, due to the financial commitment required to issue the required number of orders.

References

[1] Shamir, Adi (1979). "How to share a secret". Communications of the ACM 22 (11): 612–613.

[2] Asharov, Gilad & Lindell, Yehuda. (2011). A Full Proof of the BGW Protocol for Perfectly-Secure Multiparty Computation. Electronic Colloquium on Computational Complexity (ECCC). 18. 36. 10.1007/s00145-015-9214-4.

[3] Gennaro, Rosario & O. Rabin, Michael. (1998). Simplified VSS and Fast-track Multiparty Computations with Applications to Threshold Cryptography. Proc. of 17th PODC. . 10.1145/277697.277716.

[4] O. Goldreich, S. Micali & A. Wigderson. "How to play any mental game or a completeness theorem for protocols with honest majority". STOC 1987.

[5] Blakley, G. R. (1979). "Safeguarding cryptographic keys". Proceedings of the National Computer Conference 48: 313–317.

[6] Seth Gilbert and Nancy Lynch, "Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services", ACM SIGACT News, Volume 33 Issue 2 (2002), pg. 51–59.

[7] An introduction to Dark Pools, “lvis Picardo”.