Oracle Swap Exchange Whitepaper

Iván Diaz* and Martín Villagra*

E-mail: indexgame@gmail.com; mvillagra0@gmail.com

November 27, 2021

Abstract

Oracle Swap Exchange is a DeFi platform that enables cross-chain P2P (peer-to-peer) exchanges where trust is only weighted at blockchain-explorer oracles. It allows users to securely exchange tokens between blockchains with smart contract capabilities such as Ethereum and non-smart contract enabled blockchains such as Monero, without the need of a centralized exchange or wallet.

Keywords: Blockchain · Atomic Swap · Polygon Network (MATIC) · Monero · Oracles · Decentralized · Token · Exchange · KYC-less · DeFi · P2P Protocol

1 Motivation

With the surge of new blockchains, interoperability between them is fundamental. Atomic swaps enable the exchange of one cryptocurrency for another without using centralized intermediaries. Currently, building a bridge between almost any two blockchains is possible. However, this is not as simple for some chains like Monero because of its limited scripting capabilities. In this work, we throw light on a protocol that uses Oracles to procure atomic swaps. Our main target audience will be people looking to trade XMR (Monero's[1] blockchain cryptocurrency) in a trustless decentralized way, without the need to pass through a KYC exchange. One important aspect is that this protocol can be adapted between any two blockchains given that at least one of them supports smart contracts. We will do an extensive review of the protocol details, discuss its limitations and compare it to the state-of-the-art.

2 Background

Since the beginning of Bitcoin in 2008,[2] many other cryptocurrencies have been introduced. This technology has since then evolved into an enormous financial market. Cryptocurrencies are traded against fiat (e.g. USD, AUD, EUR) or against each other. However, due to the lack of interoperability between different blockchains, most of the trades are executed on centralized exchanges. Due to regulations, these centralized exchanges have integrated complex KYC (Know Your Customer) procedures where traders must go through lengthy processes to prove their identity. In addition, traders give up control over their hard-earned coins by depositing them in the exchange so that they can execute trades. In this case, the trader has to trust the exchange to manage their funds according to the highest standards, to protect them against thieves or avoid losing them altogether. This trust was exploited more than once in the past and billions of dollars in user funds have been lost[3] . One could say that these centralized exchanges are now a relic of the past. A new era of decentralized exchanges has commenced, adhering to the core idea of Bitcoin: censorship endurance at all levels. Decentralized exchanges powered by atomic swaps, first introduced in 2015 by TierNolan,[4] can now provide guarantees in terms of security and privacy to traders.

There are several ideas for atomic swaps. The original concept uses HTLCs (Hash Time-Lock Contracts); however, it requires that the underlying blockchains support time locks and scripts to build hash locks. More recently, protocols using adaptor signatures (also referred to as Scriptless Scripts) are getting traction. One example was recently launched by the COMIT network[5] and will be addressed later on. Throughout this work, we introduce an alternative approach using Oracles whose sole purpose is to monitor one of the blockchains to confirm the swaps. This allows the protocol to be implemented even when one blockchain only supports making (auditable) transfers.

3 Token specification

We will release a token called OSE (Oracle-Swap-Exchange) that will be used for trading on the platform. We will have a single contract where all trades will occur and it can exist in multiple blockchains. We will first focus on Polygon Network (MATIC) but we plan to release the same contract in other blockchains such as ETH 2.0[6] and BSC[7] .

3.1 Allocation

The total supply will be distributed according to Figure 1.

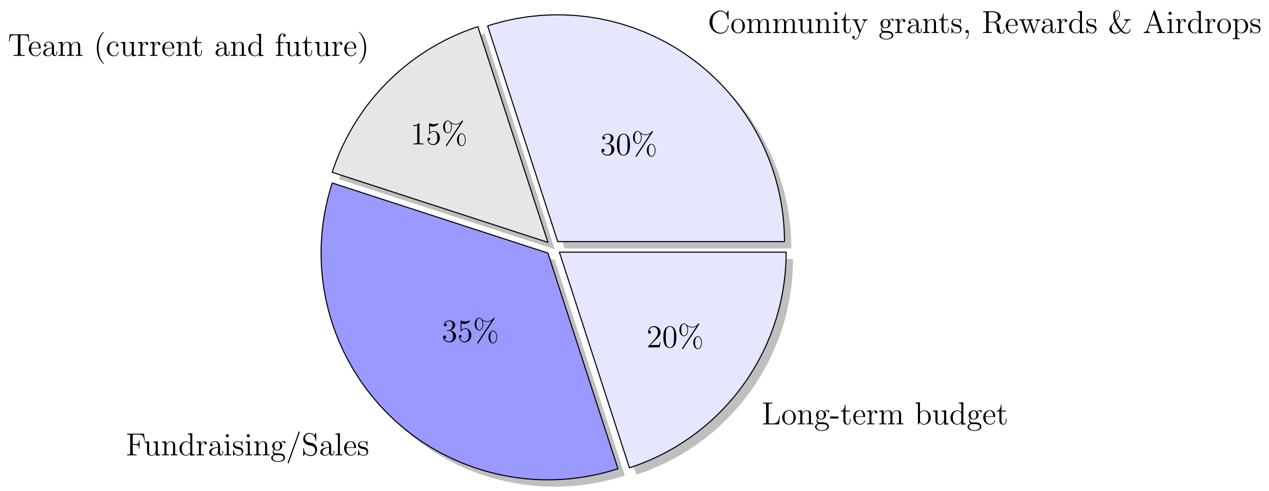

Figure 1: Token distribution

3.2 Launching

The specifics of the launch, including any ICO/IDO will be announced through our website and social media [1].

[1]https://twitter.com/OracleSwap

3.3 Rewards

Besides airdrops, during the first stages of the project we plan to give tokens to oracles and publishers as rewards for using the platform. The specifics will be announced through our social media.

3.4 Governance

During the first stages of development the team will retain control of the token but eventually once the project is mature enough the power will be delegated to the community of the stakeholders. This will be done using the OpenZeppelin ERC20Votes extension,[8] Governor contracts[9] and the Tally platform.[10]

4 OSE to XMR Protocol

Our protocol involves three different actors: Oracles, Bob and Alice. They will interact through a contract, which resides in the BSC chain and will be referred to as the "swapping contract". Both Bob and Alice have BSC and XMR addresses.

4.1 Situation

Let's suppose, for our example, that Alice and Bob have agreed to a trade in which Alice will send XMR to Bob, and Bob will send OSE to Alice. They require this exchange to be atomic, i.e. the change of ownership of one asset should effectively imply the change of ownership of the other. Additionally, should the exchange not come to fruition, they expect any committed assets to be returned to them. The whole process is illustrated in Figure 2.

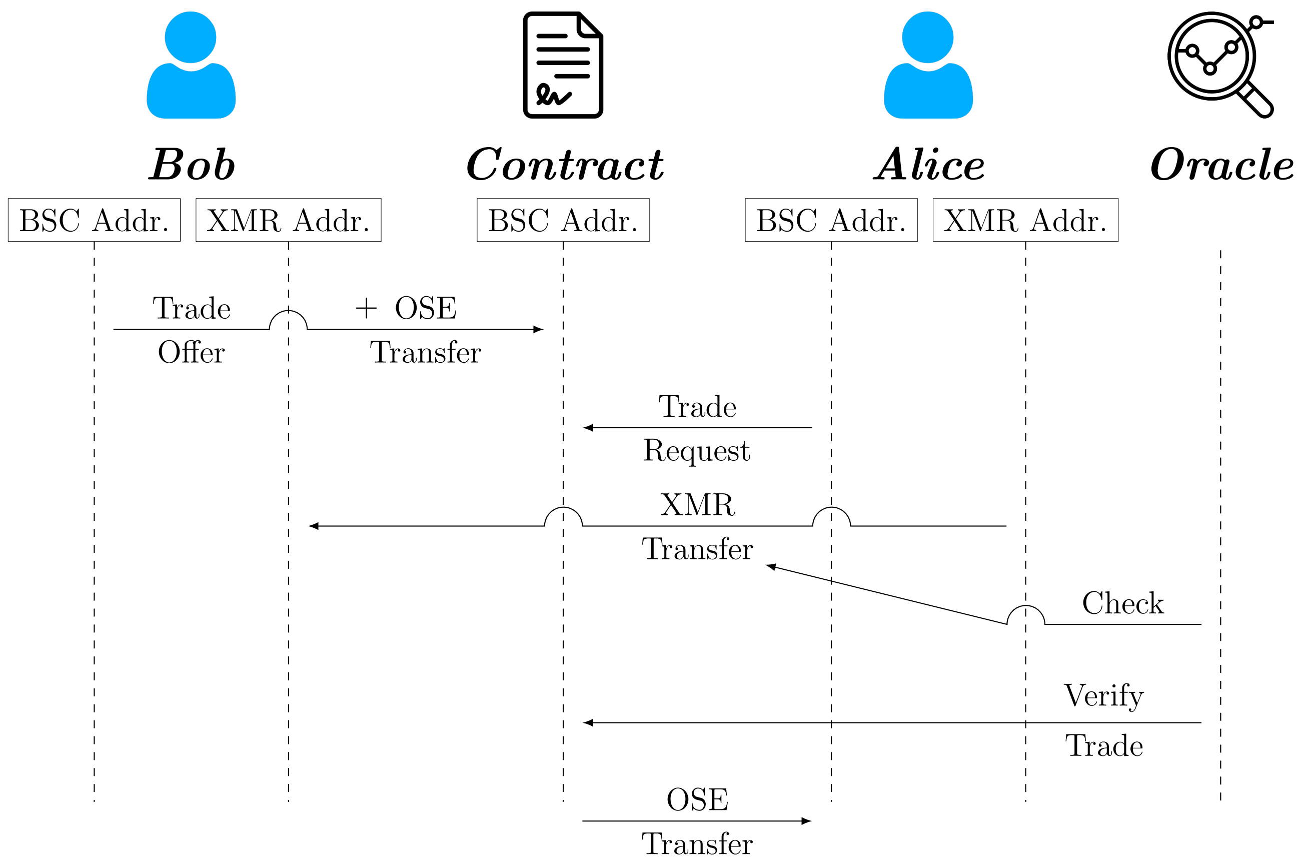

Figure 2: Diagram showing each step of the protocol that occurs when Bob wants to get XMR from Alice by giving OSE.

4.2 Bob steps

Bob will deposit OSE to the contract and generate a trade offer (publication), which effectively locks the tokens. It will specify what oracle it will use, its exchange rate of OSE/XMR, trade limits, minimal confirmations, maximum waiting time. Additionally, they will specify their XMR address to receive the requested tokens. Bob will then have to wait until Alice accepts the trade and sends the requested XMR. Once the contract receives the confirmation signed by the oracle, it will release the locked OSE to Alice. Note that Bob doesn't need to do anything after placing the publication.

4.3 Alice steps

Alice will review Bob's publication and after agreeing with the stipulated exchange rate and parameters she will send a trade request. She will then send the required XMR to Bob's Monero address. After that transaction is confirmed, she will send a trade confirmation to the contract containing the transaction. Due to Monero's privacy design she also needs to attach a transaction key. With this key the Oracle will be able to validate the transaction and will communicate this with a signed transaction to the contract. In response, the contract will release the agreed OSE to Alice.

4.4 Oracles

An oracle is identified by a BSC address. They need to be registered in the contract beforehand. They can only sign confirmed trades which have specified to use their address in their publication. To do this, once they detect a trade confirmation in the contract, they extract the information about the transaction in the Monero network and if the transaction actually happened and the destination address and amount is correct they place a transaction in the swapping contract, which causes the OSE to be released to Alice.

While an oracle can be a person manually verifying the transactions, in practice it will be implemented as bots or nodes in a decentralized Oracle system. Special care needs to be taken by Alice and Bob to choose only trusted Oracles; this will be expanded in Section 9.1.

5 XMR to OSE Protocol

That protocol described is appropriate for a use case in which the Service Provider (SP) is in the role of Alice, i.e. she offers buying XMR for OSE to her customers. However, using that protocol to swap in the opposite direction is not straight-forward i.e. an SP should not offer buying OSE for XMR. The reason for this is that even though the parties can agree off-chain on the swap, an SP (in the role of Bob) could be easily attacked: the taker (in the role of Alice) could agree on a trade, make him lock up funds on OSE and then never send the trade request. The SP could always wait for someone else to agree on the trade after some time, but that cannot be guaranteed if the agreed price differs greatly from the market price or if the volume is low. In such case Bob would be forced to remove/update his publication incurring in gas fees, while the attacker would not receive any penalties. A similar issue is encounter in the proposed atomic swaps between Bitcoin and Monero,[5] but their solution depends on Monero's ring signature scheme, which is a work-in-progress.

One possible solution to this is to move Alice publication on-chain by adding some extra steps to the protocol, this will be added in future versions of the protocol.

6 Incentives

6.1 Oracle fees

When oracles register, they need to specify a fee in OSE that they will charge for each trade. This will be paid by the seller. An oracle can update the fee, but it will pause the publications that depend on them.

7 Defences

We will need to have some security mechanisms to ensure incentives are properly balanced and reduce the risk of oracle bias and some possible attacks.

7.1 Locked trade amount

When Bob opens a publication, he deposits some OSE so buyers can trade it for XMR. When Alice is ready to buy some OSE from such publication, it needs to inform the contract that she will do that, this is because the contract needs to lock those funds to avoid conflict between multiple buyers. The funds will be locked for some time limit (waiting for payment) and they will be released to Alice when he presents a valid transaction signed by the oracle. There's a vulnerability to lock-liquidity attacks, where Alice could lock all liquidity and not trade, taking advantage of "waiting for payment" times, costing him only the fees for that. To solve this, we will need to implement a collateral to be fulfilled by Alice. Alice can have as much collateral as he wants, but it will limit the maximum amount he can trade at the same time. If a user fails to meet the "waiting for payment" deadline of a trade, the trade will be cancelled and the buyer will lose half of the trade amount from his collateral. These collateral will be divided in two parts: one for the oracle and one for the seller (to avoid oracle lock attacks).

7.2 Publication updates

In case of sudden price volatility Bob can update fees and limits of his publication and even cancel the publication, but any pending trades will still take place with the previous parameters.

8 User Interface

An interface to make trades will be made available at `https://oracleswap.exchange/`. Once ready, non-experienced users will be able to connect their wallets, make and view publications and accept them easily. The interface will of course implement the protocol described here.

9 Limitations

9.1 Centralized Oracles

In the initial release we will create an open source code for an oracle bot. This should be easy to run and fulfil the requirements correctly. We will serve and maintain an official oracle running but anyone can register a new one. However, this will probably end up with only few oracles being trusted, thus centralizing the exchange. To solve this problem in the long-term we need a more robust oracle solution and Chainlink[11] seem to fulfil this criteria. In collaboration with Chainlink's team we plan to migrate to using fully decentralized Oracles in future versions.

9.2 XMR to OSE

Section 5 mentioned one possible solution in the case when Alice wants to start the trade. The solution has the disadvantage that once published, Alice will then have to continuously monitor her publication to check for any offers and be able to quickly send the XMR once accepted. This makes the protocol asymmetric.

9.3 Comparison with previous work

Our main contribution compared to already existing protocols is that our implementation only needs smart contracts in only one of the blockchains, not both. For example, as XMR doesn't have smart contracts, it couldn't possibly be implemented using the recent Cross-Chain Interoperability Protocol (CCIP)[12] released by Chainlink.

A special case is the recently published work on atomic swaps between Bitcoin and Monero.[5] Despite offering similar services, our implementation differs greatly. To begin with there's no off-chain exchange of data needed. This makes the negotiation less vulnerable to attacks at the expense of increasing the on-chain footprint. Additionally, since we are using smart contracts in BSC we don't require adaptor signatures or any other special cryptography technique.

References

(1) van Saberhagen, N. CryptoNote v 2.0 (Monero (XMR) whitepaper). `https://www.allcryptowhitepapers.com/Monero-Whitepaper/`, accessed: 20.05.2021.

(2) Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. 2008; `https://bitcoin.org/bitcoin.pdf`, accessed: 04.06.2021.

(3) CryptoSec: Crypto exchange hacks. 2021; `https://cryptosec.info/exchange-hacks/`, accessed: 04.06.2021.

(4) TierNolan, Atomic swaps - bitcointalk forum. 2013; `https://bitcointalk.org/index.php?topic=193281.msg2224949\#msg2224949`, accessed: 04.06.2021.

(5) Hoenisch, P.; del Pino, L. S. Atomic Swaps between Bitcoin and Monero. 2021; `https://arxiv.org/abs/2101.12332`, accessed: 01.06.2021.

(6) Ethereum Whitepaper. `https://ethereum.org/en/whitepaper/`, accessed: 25.05.2021.

(7) A Parallel Binance Chain to Enable Smart Contracts. `https://github.com/binance-chain/whitepaper/blob/master/WHITEPAPER.md`, accessed: 20.05.2021.

(8) Zeppelin, O. ERC20Votes. 2021; `https://docs.openzeppelin.com/contracts/4.x/api/token/erc20#ERC20Votes`, accessed: 08.11.2021.

(9) Governance, ERC20Votes. 2021; `https://docs.openzeppelin.com/contracts/4.x/api/governance`, accessed: 08.11.2021.

(10) Tally, Tally documentation. 2021; `https://docs.withtally.com/`, accessed: 08.11.2021.

(11) Lorenz Breidenbach, B. C. A. C. S. E. A. J. F. K. A. M. B. M. D. M. S. N. A. T. F. T. F. Z., Christian Cachin Chainlink 2.0: Next Steps in the Evolution of Decentralized Or-

acle Networks. 2021; `https://research.chain.link/whitepaper-v2.pdf`, accessed: 19.09.2021.

(12) Chainlink, Introducing the Cross-Chain Interoperability Protocol (CCIP) for Decentralized Inter-Chain Messaging and Token Movements. 2021; `https://blog.chain.link/introducing-the-cross-chain-interoperability-protocol-ccip/`, accessed: 19.09.2021.