

Swaptor Whitepaper

MARKO IVANKOVIĆ¹

¹Berry Block, marko@berryblock.io

Abstract

Peer-to-peer (P2P) swaps on blockchain have the potential to greatly benefit users by eliminating the need for intermediaries in the exchange of assets. However, the use of P2P swaps on blockchain also presents several challenges, including trust issues that must be addressed in order to ensure their success. Since there is no intermediary to oversee the exchange of assets, users must rely on the trustworthiness of the other party to the transaction. In the absence of a trusted third party, it is difficult to verify the authenticity and quality of the assets being exchanged, which can lead to disputes and losses for users.

Furthermore, P2P swaps on blockchain are subject to potential security risks, such as hacking and fraud. Since the transactions are conducted directly between users, there is a greater risk of malicious actors attempting to exploit vulnerabilities in the system. This risk is exacerbated by the fact that blockchain transactions are irreversible, meaning that users have no recourse if their assets are stolen or lost.

In this paper we describe Swaptor, a decentralized P2P exchange dapp which aims to eliminate problems mentioned above.

Contents

1	Architectural Overview	3
1.1	On-chain Architecture	3
1.1.1	Signatures	3
1.1.2	Swap types	3
1.1.3	Fees	3
1.2	Off-chain Architecture	3
1.2.1	Backend	3
1.2.2	Frontend	3
1.2.3	Database & Security	3
2	Tokenomics	3
2.1	Token Allocation	3
3	Roadmap	3

1 Architectural Overview

As most modern dapps, Swaptor’s architecture is a mix of on-chain and off-chain components:

- **On-chain:** Smart contracts
- **Off-chain:** Backend, Frontend and Database

1.1 On-chain Architecture

Smart contracts are written in Solidity, and their architecture is designed to be minimalistic. Specifically, the Swaptor contract is a singular entity whose sole purpose is to verify digital signatures and execute swaps upon successful verification. Digital signatures encode the specific terms under which a given swap is considered valid.

1.1.1 Signatures

In order to verify the signature, the smart contract must possess both the original swap elements and the signature itself. Prior to being passed as an argument to one of the *acceptSwap* functions, these elements are encoded using the *abi.encode* function. The signature is generated by creating a *keccak256* hash of the encoded arguments and signing it with the private key belonging to the seller. This method dramatically reduces the gas fees because the seller never had to transfer their assets to Swaptor contract.

1.1.2 Swap types

At present, Swaptor facilitates the swapping of ERC-721 and ERC-20 tokens, enabling the execution of any combination of swaps involving these types of tokens. In the future, Swaptor intends to extend its support to include native currency and ERC-1155 tokens.

1.1.3 Fees

To accept the swap, an address will need to pay a fixed fee of \$5 in the native currency, which is calculated using Chainlink oracles[1]. It should be noted that the fee is not fixed and may be subject to change.

1.2 Off-chain Architecture

1.2.1 Backend

1.2.2 Frontend

1.2.3 Database & Security

2 Tokenomics

2.1 Token Allocation

3 Roadmap

...

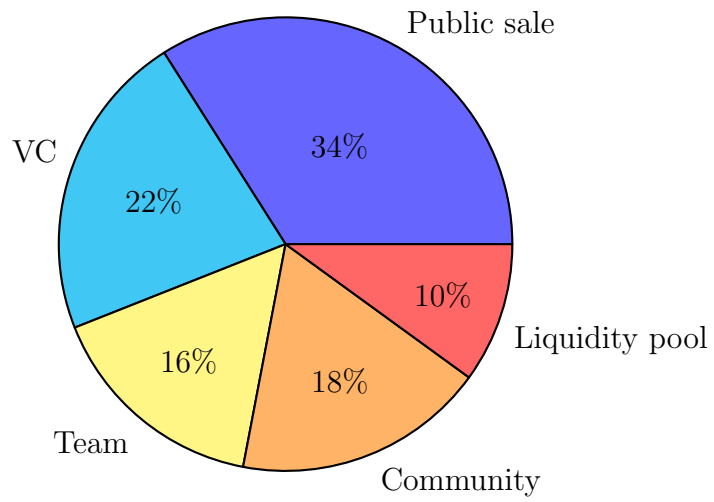


Figure 1: Pie chart showing the distribution of tokens

References

- [1] What is an oracle in blockchain? explained — chainlink, <https://chain.link/education/blockchain-oracles>