Blueshift Cross-chain Protocol

The Blueshift Team
hello@blueshift.fi

July 2023

Abstract

A solution is presented to the problem of exchanging native crypto assets between several blockchains without creation of wrapped tokens or other intermediate assets. The solution maintains consistent cross-chain token reserves using an application-specific blockchain that implements portfolio logic and serves as a settlement layer for all cross-chain operations. Cross-Chain Portfolios concept is defined. The architecture of Blues Chain - a settlement blockchain built on top of the Cosmos SDK is described. Basic cross-chain operations and consensus algorithm used by Blues Chain validators are explained. A new utility of the BLUES token is introduced as a a payment medium for the services of network validators and a reserve coin of the insurance fund.

1 Introduction

The Blueshift protocol since it has been launched proved to be an efficient and secure new DeFi instrument for both digital asset exchange and liquidity management. The protocol now embraces several blockchains, each with it's own ecosystem. But, since the ecosystems are loosely connected, the synergy between those deployments was limited. Nowadays it becomes clear that the future of blockchains is in interoperability. With these prerequisites the Blueshift team started research and development efforts in order to make the protocol cross-chain.

It appeared that the concept of Blueshift portfolios has decent advantages applicable for the cross-chain solution as well. Virtual pairs used for swaps in Blueshift portfolios can be generalized to take tokens not from one blockchain, but from different blockchains and even aggregate reserves from several blockchains for one token. Although this sounds pretty straightforward, there is still a fundamental problem that needs to be solved - maintaining token reserves consistency across several blockchains. Otherwise we can come to situations when parallel transactions in different chains can use one reserve double times or use obsolete exchange prices already modified by other transactions.

In this paper we present our solution to the problem of consistent cross-chain token reserves based on an application-specific blockchain that implements Blueshift's portfolio logic and serves as a settlement layer for all cross-chain operations. This blockchain, called Blues Chain, is not intended to store any assets - all the assets under control of the protocol stay on blockchains of their origin. Instead, Blues Chain stores consistent cross-chain states: token reserves, internal oracle prices, exponential moving averages, etc. We call all these states altogether Cross-Chain Portfolios (CCP) - a new concept introduced by Blueshift as a new cross-chain DeFi primitive.

In subsequent sections we define Cross-Chain Portfolios in detail, describe the architecture of Blues Chain and the Blueshift cross-chain protocol, basic cross-chain operations, consensus algorithm used by Blues Chain validators, new utility of BLUES token - validator staking and delegation.

2 Existing Solutions

Numerous projects are working on the problem of cross-chain interoperability. Bridges of the first generation, for example, WBTC [19] or Multichain (formerly Anyswap) [10], were based on token wrapping approach. To bridge an asset from one blockchain to another in this approach you need to send input tokens to a bridge smart contract on a source chain, where they are locked. On a destination chain a special wrapping smart contract under control of the bridge is used to mint a corresponding amount of wrapped tokens. To make a reverse bridging you send and burn wrapped tokens on the destination chain to unlock a corresponding amount of initial tokens on the source

chain. The main problem is secure and reliable cross-chain communication of proofs that the required tokens are properly locked/minted/burned/unlocked on connected blockchains. To achieve this WBTC uses a community of authorized partners (merchants and custodians) defined in a multi-signature contract governed by a DAO. Multichain uses a group of validators that apply Multi-Party Computation (MPC) signatures to approve cross-chain messages. The main disadvantage of this approach is that wrapped tokens only have value while the bridging protocol is healthy. Therefore, holders of the bridged tokens are exposed to the risk of losing their assets all the time while holding them.

This disadvantage did not allow bridges of the first generation to overcome problems of liquidity fragmentation and dispersion across several ecosystems due to their inherent technological and security risks. This is why several new solutions have emerged that are based on general message passing protocols and allow for native bridging or cross-chain swapping of assets without the need for wrapped tokens.

There are numerous general purpose cross-chain communication protocols and bridges built on top of them: Axelar [1], Router protocol [13], LayerZero [21], Stargate [14], OrbLabs/Earlybird [4], Hyperlane [8], Polymer chain [12], Gravity bridge [7], Fusion [5]. These protocols use their own validator networks to confirm and sign cross-chain messages with an additional consensus layer being used among the validators. Many of the protocols are built on top of Cosmos SDK [3] with the Tendermint/CometBFT consensus layer (Axelar, Router protocol, Polymer Chain, Gravity bridge) and IBC protocol [6] for cross-chain communication. Polymer Labs developed Polymer chain that uses zkMint - a zero-proof extension of the Tendermint consensus layer allowing for integration between Cosmos and EVM-based blockchains using Modular IBC as a cross-chain communication protocol. Fusion uses a different approach based on MPC and can be considered as an extension of Multichain solution. LayerZero, OrbLabs/Earlybird and Hyperlane represent an approach based on lightweight consensus clients deployed on connected blockchains and off-chain oracle/validator and relayer services. Stargate uses LayerZero as a communication protocol and Delta algorithm [20] to obtain instant finality of cross-chain bridging operations.

Common limitation of solutions built on top of general purpose cross-chain communication protocols is that application-specific logic is implemented on edge blockchains without the integration of the application layer on all involved chains so that they are always dependent on local protocols facing the users at the beginning and end of a transaction. This leads to lack of a consistent cross-chain state and no instant finality. Delta protocol partially solves this problem, but only for bridging of equally valued assets.

Symbiosis [17], THORChain [18] and Osmosis [11] are examples of protocols that use intermediate assets while performing cross-chain bridging or swaps.
Symbiosis is built on top of Boba BNB as a service blockchain. The cross-chain communication capabilities of Symbiosis are provided by Relayers network - a peer-to-peer (P2P) network with MPC-based signatures to approve cross-chain messages. Symbiosis uses intermediate assets (sTokens) - a type of wrapped tokens on the Boba BNB chain that represent stablecoins locked on Symbiosis smart contracts in connected blockchains (e.g. sUSDC may represent USDC on Polygon, etc.) sTokens are stored in Octopools - special liquidity pools with AMM that supports 1:1 valued assets. Symbiosis relies on third-party DEX aggregators (1inch, OpenOcean) to extend exchanging options beyond stablecoins backed by sTokens.
THORChain and Osmosis are full layer 1 AMM (automated market maker) implementations built on Cosmos SDK that allow for swapping different types of assets (not just 1:1). THORChain requires that liquidity providers create pools (pairs) with RUNE (their native token) and other assets (BTC, ETH, BNB, etc.) Then for exchanging, for example, BTC to ETH two swaps are required: BTC-RUNE, RUNE-ETH. The swap logic is implemented within Cosmos-based THORChain while special light-weight clients (Observers) monitor source blockchains for incoming liquidity flows and Signers send output transactions to destination blockchains. Osmosis has much in common with THORChain. In Osmosis their native token OSMO can also be used for pooled liquidity but this is not required. Osmosis allow for providing intermediate liquidity not only in pairs, but also in Balancer-style multi-token pools.

Intermediate assets help achieving consistency, but multihop routes are always needed to swap tokens making price slippage higher and overall op-

eration less efficient.

Another group of solutions is represented by cross-chain liquidity aggregators: Chainge [2], LI.FI [9], Swing [16]. These solutions rely on existing liquidity sources on different blockchains (decentralized exchanges and DEX aggregators) and third-party cross-chain communication protocols to perform cross-chain swaps. Also, cross-chain decentralized exchanges (cross-chain DEXes) usually act as high-level solutions, usually built on top of low-level cross-chain protocols: Osmosis DEX on top of Osmosis, THORSwap on top of THORChain, etc.

Liquidity aggregators may give better results through aggregation of different liquidity sources, but they do not add fundamental improvements to the problem of cross-chain interoperability because they rely on third-party cross-chain communication protocols.

As a result, we can conclude that using interconnected blockchains with additional consensus layer providing secure and robust cross-chain communication is the best practical approach to the problem known so far. But existing solutions do not ultimately fit all of the following criteria:

- Capital efficiency can be limited by liquidity fragmentation and dispersion as well as by the use of intermediate assets.
- Consistency can be low because the application-specific logic is often implemented on edge blockchains without integration into the communication protocol.
- Speed can be decreased by the lack of efficient consensus protocols and multihop routes.
- Decentralization can be limited by involving centralized third parties in the process of transaction validation.
- User experience is not always friendly enough compared to standard local swaps of assets.

Though existing solutions can solve these problems taken one by one, there is still demand of a consistent all-in-one solution matching all of the mentioned

criteria. And this gave motivation to launch of the Blueshift Cross-chain Protocol - an efficient portfolio-based interoperable AMM with a consistent cross-chain state without intermediate assets.

3 Starting Point: Blueshift Protocol

The Blueshift protocol [15] is the core of the represented cross-chain solution. The Blueshift protocol is a new AMM model that gives significantly better capital usage efficiency than other AMM protocols. Instead of token pairs, the Blueshift protocol has liquidity portfolios consisting of arbitrary numbers of tokens. In addition to being more capital efficient Blueshift portfolios have other advantages:

1. **Internal price oracles: flexibility of token reserves.** Unlike most other AMM protocols, Blueshift does not use token amounts to represent prices. Instead, a price for every token relative to a base token of a portfolio is stored in a special smart contract called Internal Price Oracle. This gives the protocol flexibility of managing token amounts. Every token has a corresponding target portfolio weight that can be changed by a portfolio manager without changing the token price. The protocol is able to pull real token shares up to their target weights in a decentralised way. Besides token prices internal price oracles also calculate and store exponential moving averages (EMA) of the prices. EMA values are used to enhance protocol security (see details below).

2. **Virtual pairs: low impermanent loss, better security and flexibility of AMM.** Primary target of the Blueshift protocol is allowing token swaps. Users can swap any token pair in a portfolio with the help of the Blueshift Automatic Market Maker (AMM) algorithm. Major difference of the Blueshift AMM is that the algorithm does not perform swaps directly into portfolios. Instead, the algorithm takes liquidity out of a portfolio, forms a virtual pair on-the-fly, performs a swap in this virtual pair and then returns liquidity back to the portfolio and updates the internal price oracle - all in one transaction. Virtual pairs give advantages of lower impermanent loss and better security. When a market price of some token changes, it can be adjusted by arbitrage operations. Because virtual pairs isolate token liquidity within a portfolio, the changing price of one token does not affect prices of other

tokens, causing less impermanent loss. What is even more important is that the AMM algorithm can take less liquidity for arbitrage operations in return to zero trading fees. Lower liquidity means higher price slippage for arbitrageurs. Thus, if the AMM performs arbitrage swaps with 10% of available liquidity, the resulting impermanent loss can be up to 10 times lower. Another advantage of virtual pairs is flexibility to use different AMM algorithms for different token pairs in one portfolio. For example, a different price curve can be applied for stablecoin swaps than for other token types.

3. **Protection against flash loan attacks and market fluctuations.** Using EMA values of token prices and virtual pairs makes it impossible to attack the Blueshift protocol with flash loans. In every place where price manipulations can damage the protocol security, EMA values are taken into account. EMA values in internal price oracles change only in subsequent blocks after the block that caused price modification. It means that a forged price value may not be used in the same transaction, therefore a flash loan attack will not succeed. For example, when making token deposits into portfolios, if a token price EMA is lower than a spot price, then EMA value will be used, disallowing overpriced deposits. And virtual pairs give additional level of protection. If the AMM detects that a spot token price differs from its EMA, it decreases available amounts of liquidity for virtual pairs with this token. It means that if a token price suffered manipulations, the token will have higher price slippage. Therefore, buying a token and then selling it back to the AMM will not generate zero outcome for an attacker but will cause loss - it will be simply impossible to return a flash loan. Moreover, when a token market price changes too fast, even without flash loan attacks, the same protection mechanism will prevent liquidity leakage from a portfolio - protecting liquidity providers from excessive impermanent loss.

4 Principles of Cross-chain Portfolios

As described above, Blueshift portfolios are novel DeFi technology bricks that can be considered as both financial instruments similar to ETFs and DEX liquidity reserves with high capital efficiency and low impermanent loss. In

this whitepaper we introduce another powerful concept based on the previous results - Cross-chain Portfolios.
Cross-chain portfolios (CCPs) contain several (more than two) tokens similar to their local versions, but allow liquidity provisioning from all connected blockchains. Each token may or may not be present on each connected blockchain. There may be tokens that are present on only one of the blockchains and there may be tokens that are present on several blockchains. Thus, CCPs are not single smart contracts that hold all provided tokens. Instead, they are logical views of aggregated liquidity from all the connected blockchains. Therefore, liquidity aggregation and cross-chain swaps are the most oustanding features of CCPs.

1. **Liquidity aggregation.** There are multichain tokens that can be found natively on different blockchains. There are also wrapped tokens that are produced by traditional cross-chain bridges when they move liquidity from one blockchain to another. When those tokens are used in traditional DEX liquidity pools, their liquidity is scattered across different blockchains. Blueshift CCPs allow providing liquidity of the same token on different blockchains and then using it in an aggregated form as if the token liquidity was not scattered. This is achieved by the virtual pairs feature of the Blueshift protocol. Cross-chain AMM is able to account for all available liquidity on all blockchains when creating a virtual pair for a swap. This means yet higher capital efficiency and lower price slippage.

2. **Cross-chain swaps.** Possiblity of cross-chain swaps between native tokens on different blockchains is a unique feature of Blueshift CCPs. Unlike existing solutions, a Blueshift CCP does not require any wrapping or intermediate tokens. Cross-chain swaps are performed in exactly the same way as in local portfolios - by creating a virtual pair on-the-fly and then applying an AMM algorithm for swap calculations and price updates.

3. **Token bridging.** As a special case of a cross-chain swap, Blueshift CCPs allow swaps of a token to the same nominal token on a different blockchain. In this case there are no price impact and price updates associated with the operation and this swap may be considered as a kind of bridging. An advantage over traditional bridges is that the protocol does not use wrapping of tokens - on both sides of the operation tokens

are in their native form. This solution is way more secure as the native token value does not depend on the protocol after the operation has been confirmed. Of course, this requires token liquidity to be provided in both participating blockchains.

The implementation of these principles of CCPs require maintaining of a consistent view of token amounts and prices over all the connected blockchains. This is the main purpose of the Blueshift Cross-chain Protocol and Blues Chain that are described in details in the next sections.

5 Protocol Architecture

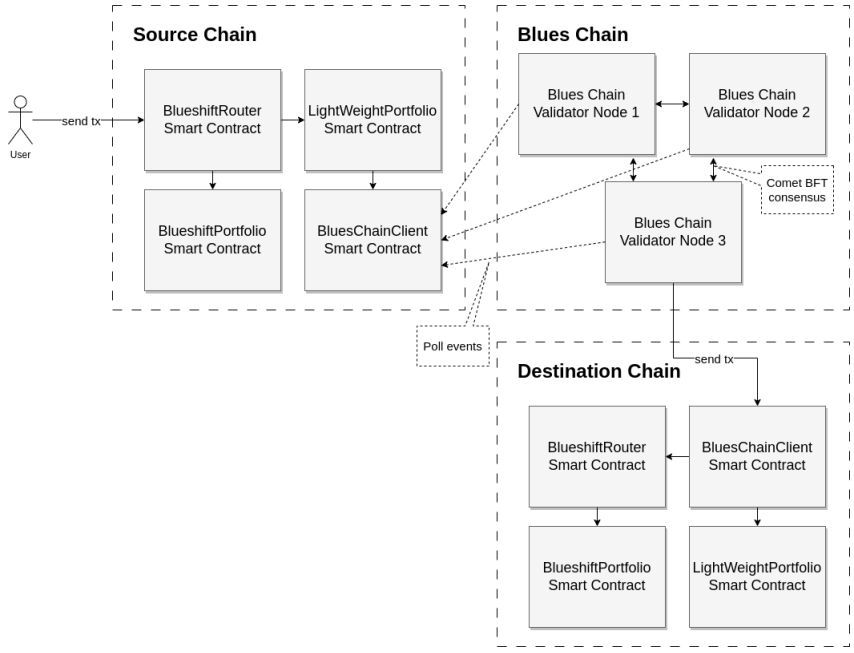

Figure 1: Blueshift Cross-chain Architecture

Blues Chain is the core of Blueshift's cross-chain protocol architecture. At the high level Blues Chain consists of several validator nodes that communicate with each other using the Cosmos CometBFT consensus protocol. Besides this every node is connected to source and destination blockchains

and continuously poll for events on the BluesChainClient smart contracts deployed in each connected blockchain.

BluesChainClient is the main smart contract supporting cross-chain operations on connected blockchains. Its purpose is to emit special events on cross-chain operations initiation and finalization. Another function of the BluesChainClient smart contract is to serve as an endpoint for transactions from Blues Chain validators.

The LightWeightPortfolio smart contract actually holds token reserves of an associated CCP on the local blockchain. It does not contain any sophisticated protocol logic. There are two functions implemented by the LightWeightPortfolio smart contract:

1. Receiving incoming tokens from users and initiating cross-chain events (actual emission of the events is performed by the BluesChainClient smart contract).

2. Sending outgoing tokens to users in response to commands passed by the BluesChainClient smart contract from Blues Chain.

Cross-chain operations can contain several hops including local swaps in standard BlueshiftPortfolio smart contracts. To achieve this the BlueshiftRouter smart contract supports mixed local/cross-chain paths. BlueshiftRouter can also be activated by the BluesChainClient smart contract when implementing several local hops as a tail of a cross-chain swap.

6 Core Cross-chain Operations

6.1 Cross-chain Fees

Blueshift cross-chain operations require additional fees paid by protocol users. Cross-chain fees are applied in addition to the existing fees in the Blueshift protocol (liquidity provider fee and protocol fee). There are two types of cross-chain fees:

1. **Validator fee.** This fee is taken as a percentage of token input amounts for cross-chain operations. The initial validator fee rate is

0.1%, this value can be changed by governance. Validator fee is deducted from input amounts before processing a cross-chain operation and then is internally swapped for BLUES. To make this possible each CCP must contain BLUES tokens. After being converted to BLUES tokens, half of the validator fee is accrued to a special account that is used to pay cross-chain validator rewards. The rest part of the validator fee is sent to the Insurance fund (see below).

2. **Gas fee.** This fee is a fixed amount that is deducted from input amounts before processing a cross-chain operation and then is internally swapped for BLUES. After being converted to BLUES tokens, gas fee is accrued to a validator that has performed a transaction in a destination blockchain as a part of a cross-chain operation to compensate for the gas amount paid by the validator. The required amount of BLUES tokens depends on gas price level in the destination blockchain. The amount of input tokens is calculated based on internal gas token and BLUES prices in the Blueshift protocol.

6.2 Cross-chain Swap

6.2.1 Route Calculation

Blueshift allows for swap routes that can contain more than one atomic swap operation (hop). Routes can include both local and cross-chain swaps in a sequence. The following route types exist:

1. Local swaps
2. Cross-chain swaps
3. Local swaps in a source chain - Cross-chain swaps
4. Cross-chain swaps - Local swaps in a destination chain
5. Local swaps in a source chain - Cross-chain swaps - Local swaps in a destination chain

Using these types of routes it is possible to swap any pair of tokens across different blockchains within Blueshift and it is not necessary to place all

tokens in CCPs - some tokens may exist in local portfolios. This feature gives the Blueshift cross-chain protocol superior flexibility.
While in the first version of the Blueshift protocol there existed the Router smart contract, but route calculations were performed outside the protocol in the DApp front-end. In the cross-chain version of the Blueshift protocol route calculation is implemented in the back-end - as a routing module inside Blues Chain nodes. The routing module applies Dijkstra algorithm to search for optimal paths that give users best possible exchange rates.

6.2.2 Elementary Swaps

During a cross-chain swap several elementary swap suboperations can occur. These elementary swaps are performed using the constant product market maker equations.
Let A be an input token amount, B - an output token amount, r_1 and r_2 - virtual token reserves according to the Blueshift protocol whitepaper [15].
Direct elemenatary swap formula:

$$B = r_2 - \frac{r_1 r_2}{r_1 + A}$$

Inverse elementary swap formula (r_2 must be greater than B):

$$A = \frac{r_1 r_2}{r_2 - B} - r_1$$

When performing a direct elementary cross-chain swap there can be a special case when the local reserve of the output token in a destination blockchain R_{L2} is less than the virtual reserve r_2. In this case we first calculate A_0 - upper limit of the input token amount that will consume all the local reserve:

$$A_0 = \frac{B \cdot r_1}{r_2 - R_{L2}}$$

If $A < A_0$, then we use the standard direct elementary swap formula, else we replace the virtual reserve with the local one:

$$B = R_{L2} - \frac{r_1 R_{L2}}{r_1 + A}$$

An inverse elementary cross-chain swap is only possible when $R_{L2} > B$ and we use the standard inverse elementary swap formula.

6.2.3 Cross-chain Swap Procedure

Let

- **amount_in** be an input token amount for a cross-chain swap;
- **amount_out** be an output token amount of a cross-chain swap;
- **LP_fee** be a liquidity provider fee (including the protocol fee), initially LP_fee = 0.3% = 0.003;
- **validator_fee** be a validator cross-chain fee amount, initially validator_fee = 0.1% = 0.001;
- **gas_fee** be a gas compensation fee amount, gas_fee = max(from_chain_gas_fee, to_chain_gas_fee), where from_chain_gas_fee and to_chain_gas_fee are current gas fee amounts for source and target blockchains respectively.

Then the procedure of a cross-chain swap goes as following:

1. Perform a direct elementary swap to buy BLUES tokens for the validator fee (sell A = validator_fee[token_in_currency] of input tokens and buy B = validator_fee[BLUES] of BLUES).
2. Perform an inverse elementary swap to buy BLUES tokens for the gas fee (buy B = gas_fee[BLUES] of BLUES for A = gas_fee[token_in_currency] of input tokens).
3. Perform a direct elementary cross-chain swap (sell A = amount_in * (1 - LP_fee - validator_fee) - gas_fee[token_in_currency] of input tokens and buy B = amount_out of output tokens.

6.3 Cross-chain Bridging

Blueshift CCPs can contain same tokens on different blockchains. In this case users can initiate a procedure similar to cross-chain swaps, but having the same token as both input and output. This special cross-chain swap is equivalent to a bridging operation and is performed using a slightly different algorithm.
Let

- **amount_in** be an input token amount for a bridging swap;
- **amount_out** be an output token amount of a bridging swap;
- **validator_fee** be a validator cross-chain fee amount, initially validator_fee = 0.1% = 0.001;
- **gas_fee** be a gas compensation fee amount, gas_fee = max(from_chain_gas_fee, to_chain_gas_fee), where from_chain_gas_fee and to_chain_gas_fee are current gas fee amounts for source and target blockchains respectively.

Then the procedure of a bridging swap goes as following:

1. Perform a direct elementary swap to buy BLUES tokens for the validator fee (sell A = validator_fee[token_in_currency] of input tokens and buy B = validator_fee[BLUES] of BLUES).
2. Perform an inverse elementary swap to buy BLUES tokens for the gas fee (buy B = gas_fee[BLUES] of BLUES for A = gas_fee[token_in_currency] of input tokens).
3. At the output we return the same amount of tokens without price impact, but withholding cross-chain fees (amount_out = amount_in * (1 - validator_fee) - gas_fee[token_in_currency], amount_out must be less than the local token reserve in the destination chain).

6.4 Cross-chain Liquidity Deposit

Providing liquidity to a Blueshift CCP is a cross-chain operation called liquidity deposit. A user can provide several tokens in any connected (source) blockchain. As a result, liquidity portfolio (LP) tokens are minted. The user can choose on which of the connected (destination) blockchains to receive the LP token amount.
Let

- **amount_in_i** be an input amount of i-th deposited token;
- **LP_out** be an output amount of LP tokens;
- **LP_fee_and_impact** be an amount of fees and price impact when the deposit operation causes portfolio imbalancing;

- **validator_fee** be a validator cross-chain fee amount, initially validator_fee = 0.1% = 0.001;
- **gas_fee** be a gas compensation fee amount, gas_fee = max(from_chain_gas_fee, to_chain_gas_fee), where from_chain_gas_fee and to_chain_gas_fee are current gas fee amounts for source and target blockchains respectively.

Then the procedure of a liquidity deposit goes as following:

1. Perform the liquidity deposit and mint the output amount of LP tokens (LP_out) according to the Blueshift Protocol whitepaper. If the deposit operation causes imbalalancing of the portfolio, LP_fee_and_impact will be applied.
2. Estimate an LP token amount required to withdraw BLUES tokens for the gas fee (gas_fee[BLUES]), get gas_fee[LP_token]. Then withdraw BLUES tokens from the portfolio.
3. Withdraw BLUES tokens for the validator_fee providing LP_out * validator_fee amount of LP tokens.
4. Send the rest of LP tokens to the user. If $LP\_out < gas\_fee[LP\_token]/(1 - validator\_fee)$, then the whole operation fails.

6.5 Cross-chain Liquidity Withdrawal

Withdrawing liquidity from a Blueshift CCP is a cross-chain operation called liquidity withdrawal. A user can provide LP tokens in any connected (source) blockchain. The user can choose on which of the connected (destination) blockchains to receive the withdrawn token amounts.
Let

- **LP_in_i** be an input amount of LP tokens to withdraw i-th of the portfolio tokens;
- **LP_fee_and_impact** be an amount of fees and price impact when the withdrawal operation causes portfolio imbalancing;
- **validator_fee** be a validator cross-chain fee amount, initially validator_fee = 0.1% = 0.001;

- **gas_fee** be a gas compensation fee amount, gas_fee = max(from_chain_gas_fee, to_chain_gas_fee), where from_chain_gas_fee and to_chain_gas_fee are current gas fee amounts for source and target blockchains respectively.

Then the procedure of a liquidity withdrawal goes as following:

1. Estimate an LP token amount required to withdraw BLUES tokens for the gas fee (gas_fee[BLUES]), get gas_fee[LP_token]. Then withdraw BLUES tokens from the portfolio.

2. Withdraw BLUES tokens for the validator_fee providing $\sum_i LP\_in_i \cdot validator\_fee$ amount of LP tokens.

3. Perform the liquidity withdrawal according to the Blueshift Protocol whitepaper. The amount of LP tokens for each token withdrawal is

$$LP\_in_i \cdot (1 - validator\_fee) - gas\_fee[LP\_token] \cdot \frac{LP\_in_i}{\sum_i LP\_in_i}$$

   If the deposit operation causes imbalalancing of the portfolio, LP_fee_and_impact will be applied.

4. Send the withdrawn tokens to the user. If $\sum_i LP\_in_i < gas\_fee[LP\_token]/(1 - validator\_fee)$, then the whole operation fails.

7 Blues Chain: A Cross-chain Settlement Network

7.1 Blues Chain Overview

Blues Chain is an application specific blockchain built on top of Cosmos SDK. The main goal of Blues Chain is holding consistent and reliable data about cross-chain token reserves and prices that constitute Blueshift CCPs. This consistency allows for direct cross-chain operations (swaps, liquidity deposits, etc.) without using intermediate assets or wrapped tokens.
Assets are not bridged to Blues Chain. Blues Chain is only responsible for accounting, cross-chain messaging and consensus. Thus, Blues Chain is a layer zero solution for the Blueshift cross-chain protocol.

Another important aspect of Blues Chain is that, being an application specific blockchain, it contains a builtin implementation of the core Blueshift protocol logic, including the AMM, internal price oracles and virtual pairs. The implementation in Go language is significantly more efficient than smart contracts executed on a virtual machine (e.g. EVM), giving better performance and scalability. This approach also helps Blueshift's expansion to different blockchain platforms in the future, because only simple lightweight portfolio smart contracts are required on target platforms. This can be also beneficial when integrating blockchains with high transaction costs as most part of the logic is implemented on Blues Chain and does not require spending lots of gas on transactions.

7.2 Transaction Lifecycle

The Blues Chain transaction lifecycle is implemented on top of CometBFT consensus algorithm which is a standard part of the Cosmos SDK.

Each cross-chain operation passes the following steps:

1. **Cross-chain operation initiation.** A users sends tokens to a lightweight portfolio smart contract on a source blockchain. Then the BluesChainClient smart contract emits the OnSend event.

2. **Source events detection.** All validator nodes listen for lightweight portfolio events in source blockchains. When a validator node detects a new event, it builds a new transaction proposal, signs it with the validator's key and adds it to the mempool.

3. **Block creation.** A designated validator node for a next Blues Chain block queries the mempool for new proposals. If it finds a proposal that has got confirmations with more than 2/3 of validators' total voting power, it selects the proposal for addition to the next block. After processing the whole mempool, the designated validator node creates and broadcasts a new block proposal.

4. **Consensus.** All Blues Chain validators perform the standard CometBFT consensus algorithm to confirm the new block. During the consensus procedure each transaction gets one of two possible statuses: Confirmed or Rollback.

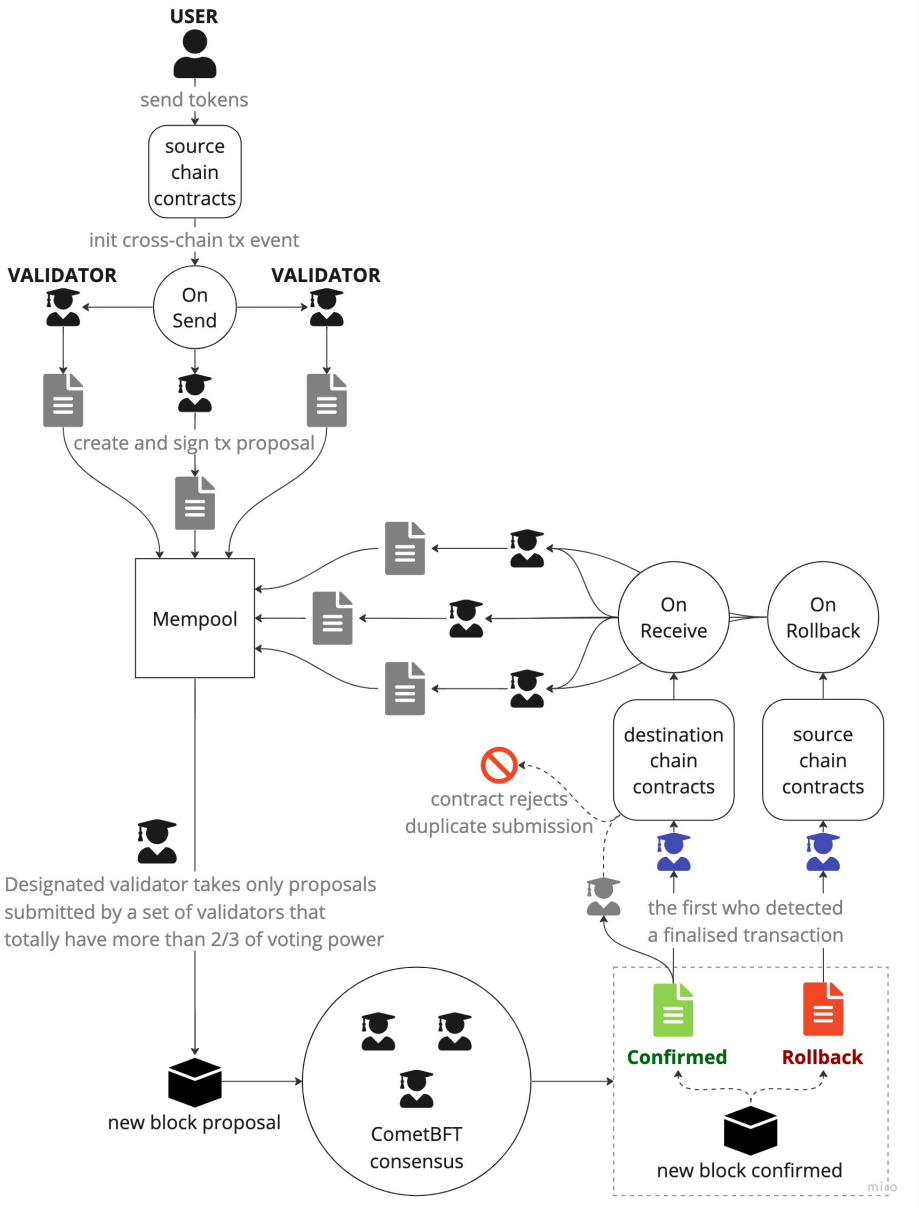

Figure 2: Blues Chain Transaction Lifecycle

5. **Transaction finalization.** When any validator node detects consensus achievement for a new transaction with the Confirmed status, it sends a finalization transaction to the BluesChainClient smart con-

tract in the destination blockchain. Then the BluesChainClient smart contract emits the OnReceive event.

6. **Transaction rollback.** When any validator node detects consensus achievement for a new transaction with the Rollback status, it sends a rollback transaction to the BluesChainClient smart contract in the source blockchain. Then the BluesChainClient smart contract emits the OnRollback event.

7. **Gas compensation.** All validators listen for lightweight portfolio events in destination blockchains. When a validator node detects a new OnReceive/OnRollback event, it builds a new transaction proposal for the gas compensation, signs it with the validator's key and adds it to the mempool. When this transaction gets confirmations with more than 2/3 of validators' total voting power, it gets selected for addition to the next block and passes the standard consensus algorithm. This way the validator who has finalized or rolled back the transaction gets the gas compensation fee (in BLUES).

7.3 Validator Staking and Delegation

Blues Chain relies on staking and validation logic of Cosmos SDK. To become a validator, one must stake BLUES tokens. The staking mechanism is based on the existing Blueshift staking smart contracts (yield pools). After staking in the yield pools, users will be able to delegate their stakes to existing validators or to themselves.

Validators and delegators receive rewards - part of the validator fee proportional to their staked BLUES amount (voting power). Validators can set additional fee that they take from their delegators' rewards in addition to their own rewards. The rewards in BLUES are accrued to validators / delegators accounts on Blues Chain and can be claimed in the yield pools.

Delegators can cancel or change their delegations. The cancellations do not take immediate effect, but are subject to unbinding period.

7.4 Insurance Fund

Cross-chain operations carry additional risks compared to standard blockchain transactions because states of different blockchains are not guaranteed to be consistent. To overcome this fundamental problem Blueshift introduces a special insurance fund. The fund is filled by withholding 50% of the validator fee that is paid in BLUES tokens by users when performing cross-chain operations. The fund is controlled by Blues Chain and can only be used for reimbursements in case of occasional losses. If no incidents of loss occur then tokens will be locked in the insurance fund forever.

8 Source and Destination Blockchain Connectors

8.1 Overview

Connectors to source and destination blockchains are essential parts of the Blueshift Cross-chain Protocol. Each validator node must be connected to all supported blockchains in order to initiate and confirm cross-chain operations. These connections are implemented as standalone applications - Validator Client Modules. There could be different types of the validator client modules: EVM-compatible modules, Cardano modules and so on, the number of module types will grow as Blues Chain adds support to different blockchain types.

8.2 Validator Client Modules

The validator client modules implement two major functions:

- Listen to special events in source and target blockchains and transmit events data to the Blues Chain node.
- Send finalization or rollback transactions to destination and source blockchains on behalf of validators.

8.3 Blues Chain Client Events

The events that are detected by the validator client modules are generated by the Blues Chain client smart contracts. There are three general types of

the Blues Chain client events:

- **OnSend event** is emitted when a cross-chain operation is initiated in a source blockchain.
- **OnReceive event** is emitted when a cross-chain operation is completed in a destination blockchain.
- **OnRollback event** is emitted when a cross-chain operation is rolled back in a source blockchain.

All the events carry data packets. The data packets have general fields as well as arbitrary data depending on a cross-chain operation type. The general data packet fields are:

- **dataType** is a cross-chain operation type (SWAP, DEPOSIT or WITHDRAW).
- **sourceChain** is a source blockchain id.
- **destinationChain** is a destination blockchain id.
- **serialNumber** is a number of the event (each cross-chain operation type has it's own sequence of event numbers).
- **portfolioId** is a cross-chain portfolio (CCP) id.
- **sender** is an account (wallet) address of sender in the source blockchain.
- **receiver** is an account (wallet) address of receiver in the destination blockchain.

8.4 Signatures

The Blues Chain client smart contracts require that each cross-chain operation is properly signed by Blues Chain validators. To achieve this goal the smart contracts store a hash value of the current set of validators' account addresses in the current blockchain - a valset. The initial valset is defined at the smart contract deployment and later on it can be changed only by a special transaction signed by the current (old) valset.

As a part of transaction lifecycle Blues Chain nodes gather validator signatures compatible with the destination blockchain in every transaction. When a validator client module executes the transaction in the destination blockchain it must supply the current valset and all the collected signatures to the Blues Chain client smart contract.

Upon receiving a transaction from a validator client module, the Blues Chain client smart contract first calculates and checks the hash value of the valset. If the valset is unchanged, the smart contract proceeds to checking the validators' signatures. To save computations and blockchain gas spending the signatures must be sorted by validator voting power descending. Thus, the smart contract does not need to process all the supplied signatures, it can stop checking when accumulated voting power exceeds 2/3 of the total voting power.

9 New Utility for BLUES Token

With the launch of Blues Chain BLUES becomes a payment medium for the services of network validators. Every cross-chain transaction requires a small amount of BLUES to be paid (the validator fee plus the gas fee). Users will not need to have BLUES beforehand. The fees are taken from input token amounts and then internally swapped to BLUES. This is why all CCPs must contain BLUES.
Another important new utility of BLUES is the validator's staking. The validator's staked amount defines their voting power and share of the validation rewards. The staked amount is also used to retain slashes for validator's malfunction.
And, finally, we introduce the Blueshift insurance fund as an additional utility for BLUES tokens. The insurance fund accumulates 50% of all cross-chain validator fees taken from protocol users. These fees, nominated in BLUES tokens, are locked in lightweight portfolio smart contracts under control of Blues Chain and are unavailable for swaps and other operations. The purpose of these locked assets is to guarantee repayments of losses should they occur during the protocol operation. As these circumstances have very low probability, BLUES tokens will be locked in the insurance fund virtually forever.

10 Summary

In this whitepaper we have presented the Blueshift cross-chain protocol that can be used to exchange native crypto assets between several blockchains without creation of wrapped tokens or other intermediate assets. We also introduced a concept of cross-chain portfolios (CCP) as a generalization of Blueshift portfolios that have the same swap and liquidity management logic, but allow for aggregating assets from different blockchains in one logical portfolio.

The second major part of this whitepaper is the description of Blues Chain - an application-specific blockchain built on top of the Cosmos SDK and used as a common settlement layer for all Blueshift's cross-chain operations. We have presented the architecture, functional description and transaction lifecycle of Blues Chain.

In the third part we explained source and destination blockchain connectors as well as validator staking and delegation logic. We described a new important utility of the BLUES token as a a payment medium for the services of network validators and a reserve coin of the insurance fund.

References

[1] Axelar network: Connecting applications with blockchain ecosystems. `https://axelar.network/axelar_whitepaper.pdf`.

[2] Chainge finance. `https://chainge-finance.gitbook.io/chainge-finance/discover-chainge/about-chainge`.

[3] Cosmos sdk documentation. `https://docs.cosmos.network/main`.

[4] Earlybird protocol. `https://docs.earlybird.xyz/earlybird/readme`.

[5] FUSION FOUNDATION. Fusion whitepaper. an inclusive cryptofinance platform based on blockchain. `https://www.fusion.org/themes/fusion/assets/pdf/Fusion-White-Paper.pdf`.

[6] Christopher Goes. The interblockchain communication protocol: An overview. https://github.com/cosmos/ibc/raw/old/papers/2020-05/build/paper.pdf.

[7] Gravity bridge. https://www.gravitybridge.net/.

[8] Hyperlane. https://docs.hyperlane.xyz/docs/introduction/readme.

[9] Li.fi documentation. https://docs.li.fi/.

[10] Multichain - cross-chain router protocol. https://multichain.xyz/.

[11] Osmosis docs. https://docs.osmosis.zone/.

[12] Polymer: The ibc transport hub. https://www.polymerlabs.org/.

[13] Router chain. https://www.routerprotocol.com/router-chain-whitepaper.pdf.

[14] Stargate. https://stargate.finance/.

[15] Igor Struchkov, Igor Mikhalev, and Alexey Lukashin. Blueshift protocol. https://blueshift.fi/wp-content/uploads/2023/02/blueshift-whitepaper.pdf.

[16] Swing.xyz — cross-chain crypto & bridge aggregator. https://swing.xyz/.

[17] Symbiosis documentation. https://docs.symbiosis.finance/.

[18] Thorchain: A decentralised liquidity network. https://github.com/thorchain/Resources/blob/master/Whitepapers/THORChain-Whitepaper-May2020.pdf.

[19] Wrapped tokens. a multi-institutional framework for tokenizing any asset. https://wbtc.network/assets/wrapped-tokens-whitepaper.pdf.

[20] Ryan Zarick, Bryan Pellegrino, and Caleb Banister. Delta: Solving the bridging trilemma. https://www.dropbox.com/s/gf3606jedromp61/Delta-Solving.The.Bridging-Trilemma.pdf.

[21] Ryan Zarick, Bryan Pellegrino, and Caleb Banister. Layerzero: Trustless omnichain interoperability protocol. https://layerzero.network/pdf/LayerZero_Whitepaper_Release.pdf.

Disclaimer

This paper is for general information purposes only. It does not constitute investment advice or a recommendation or solicitation to buy or sell any investment and should not be used in the evaluation of the merits of making any investment decision. It should not be relied upon for accounting, legal or tax advice or investment recommendations.