Swim: Seamless Cross-Chain Liquidity

July 2021

Abstract

The Swim Protocol enables cross-chain swaps by establishing liquidity pools of two or more tokens across multiple blockchains. Our concept is based on existing designs from stableswap AMMs, but introduces a novel cross-chain component using Solana-based bridge Wormhole. Our approach reduces the current barriers faced by participants when performing cross-chain transactions, ultimately enabling true interoperability between various blockchain networks.

1 Introduction

1.1 The Multi-Chain World

Decentralized financial protocols and smart contract-enabled blockchains have grown exponentially in the past two years. The total value of assets being held on these protocols has reached $145 billion compared to $1 billion at the beginning of 2020.[1] Though the Ethereum network accounted for virtually all activity and assets in decentralized finance in its embryonic stages, the situation today is markedly different. Spurred by the relatively high costs and slow speeds to transact on Ethereum, a significant portion of user activity and assets have migrated over to other blockchains.[1]

Chief among these non-Ethereum chains is Binance Smart Chain. The network was launched in late 2020, and has since grown to become the second largest smart contract chain. Another rapidly growing blockchain is Solana; due to its advantages, it has grown to become one of the largest smart contract networks in terms of total value locked, with $1.6 billion in total value locked as of August 2021.[1]

It is clear that DeFi has begun to grow beyond Ethereum towards an increasingly multi-chain paradigm. However, the infrastructure supporting cross-chain financial transactions has proven to be lacking. Exploits of high profile cross-chain platforms only serve to highlight the difficulty and importance of building highly secure cross-chain technologies.[2][3][4][5]

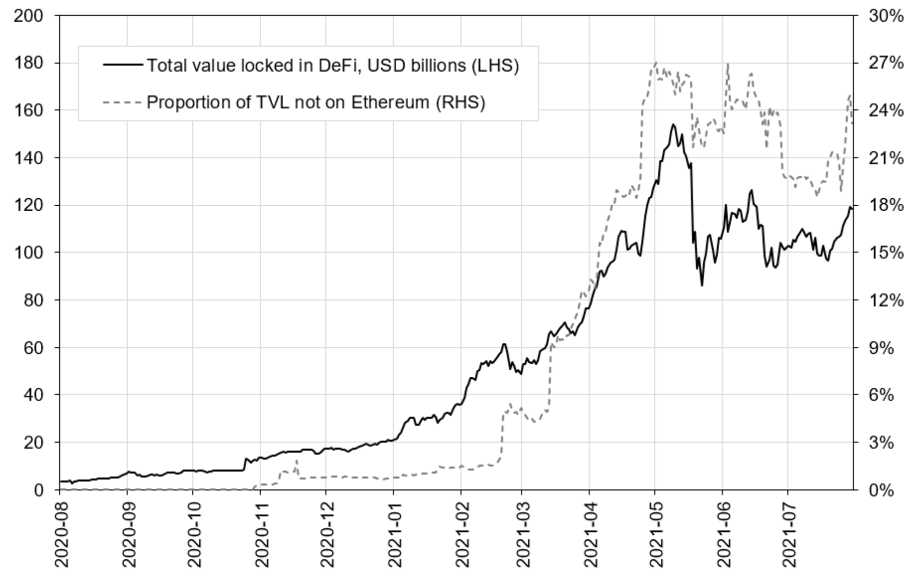

Figure 1: DeFi TVL and proportion of TVL not on Ethereum[1]

The ability to move assets between chains is paramount within a multi-chain ecosystem where each blockchain sees significant user volume and asset flows. In the absence of quick and efficient means of performing more complex cross-chain actions, market participants must rely on existing, inefficient methods to bridge assets. In our view, users currently face the following issues:

- *Bridged asset compatibility:* Most cross-chain bridges create a proprietary wrapped version of each successfully bridged token. An example of this is ether bridged from Ethereum to Binance Smart Chain. This ETH becomes BEP20 wrapped ether. When this wrapped ether is then further bridged to Solana, the resulting wrapped ether becomes non-fungible with ether bridged directly from Ethereum to Solana.

- *Reliance on centralized exchanges:* In many instances, the most efficient method to move assets from one chain to another is via a centralized exchange. The user deposits an asset from one chain only to withdraw it to another, relying on the CEX to be an intermediary. This poses obvious barriers for a number of participants, including those without accounts at the relevant exchange or those who prefer to use on-chain services, in addition to the myriad risks posed by centralization such as custodial risk and potential for censorship, to name a few.

- *Longer wait times:* Traditional token bridges, which are predominantly designed with a proof-of-authority approach, are relatively slow. 15-minute delays are common among cross-chain bridges, and in some cases even up to week-long withdrawal periods.[6] The resulting experience for users is suboptimal and needlessly cumbersome.

- *Transaction limits:* Some bridges restrict cross-chain activity with transaction limits, arbitrarily hindering users' ability to move freely between chains. These measures do improve security since misappropriated assets are forced to remain on-chain, but are only as effective as the network's ability to control and censor activity. This is antithetical to the principles of DeFi, in our view.

- *Third party wallet and tokens:* Some cross-chain solutions require the use of a separate wallet and a separate token native to that platform.[7][8] Recent exploits, however, point towards potential issues with this kind of cross-chain architecture, not to mention the capital inefficiency and complexity introduced.

When there are so many barriers involved in cross-chain transactions, it can be surmised that many market participants would simply not bother at all, choosing to remain primarily on a single chain. The end result that we have seen empirically is that networks become siloed from one another. Furthermore, assets from one blockchain have limited use in another.

Swim aims to solve the issues described above. Our goal is to facilitate the fluid movement of assets between major blockchain networks, enabling a free and efficient multi-chain ecosystem to thrive.

1.2 Cross-Chain Token Bridges

Token bridges have existed for some time as there has always been a need for cross-chain compatibility. More recently, newer bridging protocols have appeared with the primary goal of facilitating cross-chain swaps, especially with Ethereum.[7][8] Various blockchains have also developed their own respective bridges interfacing with Ethereum. This section lays out existing bridging solutions and variants.

1.2.1 Proof of Authority (PoA) Bridges

Proof of Authority bridges rely on a centralized, trusted set of validators, or authorities, to ensure that the tokens locked on the origin chain match the bridged assets that are minted on the destination chain. Unlike a fully decentralized validation scheme, where anyone can run a node, validators in a PoA node are known, whitelisted parties. Since there are typically much fewer nodes involved, these kinds of bridges are typically much more efficient than in fully trustless systems.

It should be clear that these efficiency gains come at a cost. The integrity of the bridge is entirely dependent on the trustworthiness of its chosen set of authorities, which means that the system can have a higher potential of failure. On a related note, validators can often be politically aligned. This means validators can be influenced or coerced into performing censorship or some other action detrimental to bridge users.

1.2.2 Hash Time Locked Contracts (HTLC)

Hash Time Locked Contracts (HTLCs) involves the uses of hashlocks and timelocks, and enables the exchange of tokens across different blockchains without the need of a middle man. The steps are as follows:

1. Alex initiates the transaction by generating a secret key, also known as a preimage, to be used later onin the transaction.

2. Alex deploys a smart contract on Chain A, which will only send the tokens to Bob if the secret generated by Alex is revealed. If the transaction does not occur within a predefined time, the assets will be released back to Alex.

3. Alex locks tokens into the HTLC on Chain A and generates a hash corresponding to the secret, revealing this hash to Bob.

4. Bob deploys a HTLC on Chain B using the hash from Alex. He locks tokens into the contract and designates Alice as the recipient.

5. Since Bob's HTLC can only be unlocked with Alex's secret, Alex uses the secret to claim the tokens in Bob's HTLC on Chain B.

6. Alex has now revealed the secret preimage to Bob, and uses it to claim Alex's payments on Chain A.
7. Alex now owns tokens on Chain B, while Bob owns tokens on Chain A. The cross-chain swap is complete.

HTLCs have several drawbacks that have been extensively researched, including the free option problem, griefing attacks, and the complexity involved in doing a cross-chain swap.[9]

1.2.3 Conditional Transfers

Conditional transfers are similar to HTLC-based swaps, but do not rely on a secret preimage and a hash. Instead, each successive step in a transaction is dependent on the outcome of previous steps, with observer nodes or validators sitting in between to relay the outcome of actions across chains. This means that conditional transfers are less secure in comparison to atomic swaps or HTLCs.

The complexity involved in conditional transfers can rise quickly as assets are locked in multiple contracts with a one-way state peg maintaining states across chains. This enlarges the surface area of attack and can result in looping attacks if not maintained properly.[5]

1.2.4 Limitations of Wrapped Tokens

Most cross-chain token bridges are not bridges in the true sense of the word. Instead, they lock the asset on its native chain, only to mint a representation of it on the mirroring chain. In our view, however, a cross-chain token bridge should allow users to receive the canonical, or most commonly used, form of that token, and not a proprietary wrapped version that may or may not be canonical. Our goal is to build such a protocol, paving the way for true interoperability between all blockchains.

1.3 Automated Market Makers

There have been many variants of the Automated Market Maker (AMM) both in and outside of cryptocurrency markets. An AMM establishes a relationship between trade size, market depth, and price. Market scoring techniques like the logarithmic market scoring rule (LMSR) were first implemented in prediction markets.[10]

Building on this work, decentralized AMMs were created using smart contracts on public blockchains like Ethereum. Most notable is a large family of AMM called the constant factor market maker (CFMM), whereby liquidity pools are established and an arbitrary invariant is used to determine prices and slippage at each price point and trade size. By design, CFMMs have the ability to provide at least some liquidity at any given price point, for any given trade size.

Uniswap v2, for example, is a constant product market maker, defining an invariant k which governs the relationship between the quantities of one token and another, x and y respectively, where $x * y = k$. The resulting price slippage curve is designed to provide acceptable slippage levels for reasonable trade sizes across a wide range of prices, allowing Uniswap to serve as a generalized AMM for all asset pairs, especially volatile ones.[11]

Notwithstanding the benefits of Uniswap's generalized constant product mechanism, this approach falls short when it comes to token pairs that are relatively stable or have ample liquidity. Users do not benefit from an AMM that offers ample liquidity at all price points for token pairs whose exchange rate is expected to have minimal fluctuations from a set rate. A prime example is stablecoin pairs or pairs of a base asset and its derivatives, which should trade at or near parity the vast majority of the time. In this way, generalized CFMMs of the type above that are designed for volatile assets would quickly become inefficient for these stable assets, since traders would incur too much slippage when trading these assets.

The solution for this is to alter the CFMM invariant formula such that the slippage is exceedingly low at or around certain price points, with slippage increasing at a much higher rate outside of these bands. Owing to their ability to offer much better prices for highly correlated assets, such AMMs have proven to be superior when trading between stablecoins, leading to widespread adoption. Swim is adopting this mechanism for its simplicity and generalizability, which we detail in the sections below.

2 Swim Protocol

Many current cross-chain AMM implementations introduce wrapped assets. The resulting issues are two-fold. Firstly, the user will often receive non-canonical forms of their token, which requires the user to perform additional actions before their assets can be used. A related but distinct problem that arises is that liquidity for certain assets becomes fragmented into multiple versions of the same asset.

Swim sets out to solve these issues by combining AMM technologies with Wormhole's bridging functionality. When built on Solana with its high throughput and subsecond block finality, the resulting cross-chain AMM enables native asset cross-chain swaps in a seamless and efficient manner.

2.1 Wormhole

Wormhole is an example of a PoA token bridge, maintaining a number of cross-chain node operators that verify the authenticity of bridging activity. According to Wormhole:

> [Wormhole] uses decentralized cross-chain oracles — called guardians — operated by a set of node operators that include top Solana validators and other ecosystem stakeholders whose incentives are strongly aligned with Solana and Serum. Those guardians certify token lockups and burns on one chain in order to mint new tokens or release tokens on the other, and vice versa.[12]

Although there are relatively few guardians responsible for validating transactions, we note that guardians' incentives are aligned with the Solana ecosystem. Wormhole will likely be the most performant token bridge in terms of speed, while maintaining a robust degree of security.

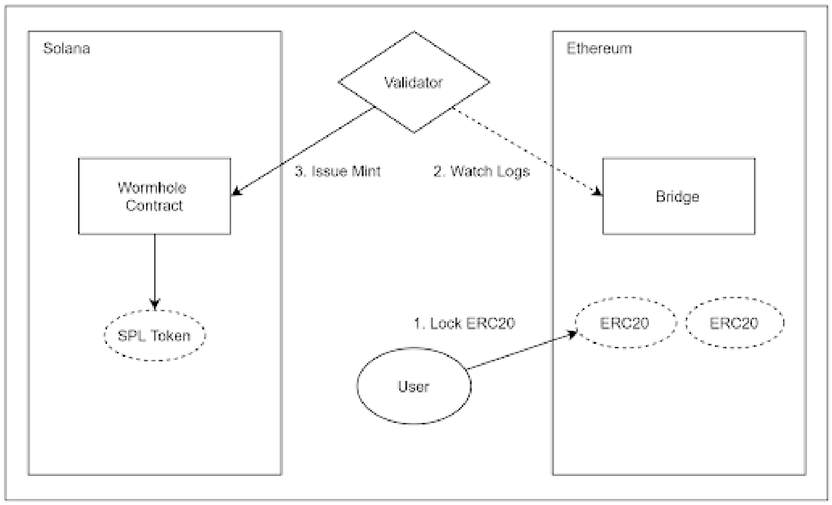

Figure 2: Using Wormhole to bridge from Ethereum to Solana

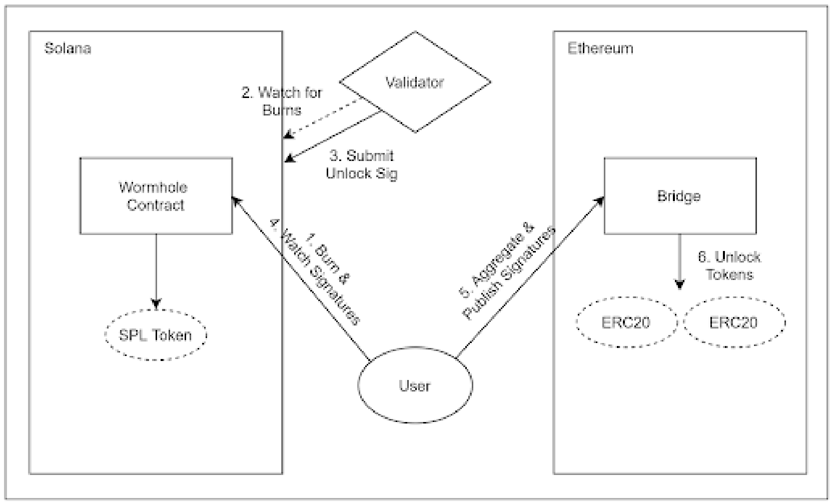

Figure 3: Using Wormhole to bridge from Solana back to Ethereum

The initial implementation of Swim utilizes the current version of Wormhole, relying on its standard mint and burn bridging mechanism, which creates wrapped assets when bridged to Solana. Future iterations of Wormhole could augment the design of Swim Protocol. Transmission of generic messages and data can allow Swim to utilize parked assets rather than locking them in the Wormhole contract. On a related note, upgrades to Wormhole are anticipated to support alternate blockchains such as Binance Smart Chains and Terra, which will further improve Swim's functionality.

2.2 Multi-Chain Liquidity Pools

As assets are deposited onto Swim's Ethereum smart contracts, the tokens will be bridged via Wormhole to Solana. The resulting wrapped assets on Solana will be deposited into smart contracts on Solana, which are responsible for maintaining an accurate state of token balances in the pool.

In order to maintain a seamless user experience, the wrapped assets are abstracted away from the front end. Future updates to Wormhole will enable protocols to send generic messages between blockchains, which could potentially remove the need for wrapped assets.

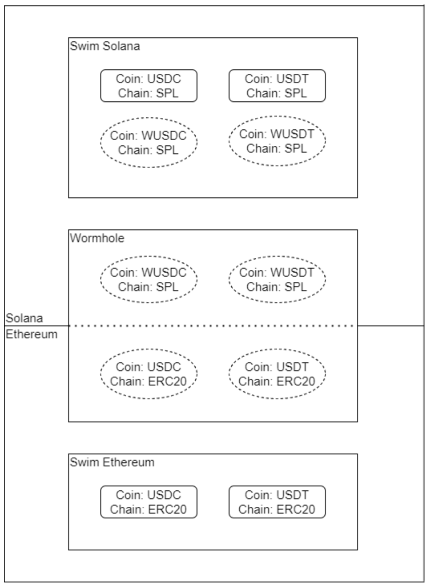

Figure 4: Basic architecture of Swim liquidity pools

The primary motivations for this design are twofold: Solana's high speed and low cost keeps protocol-level transaction fees to a minimum. Furthermore, performing all computations on Solana in sequence greatly reduces race conditions. As transactions are performed in the order they were received, all uncertainty about the state of the liquidity pools is removed. In this way, an accurate transaction price and quantity can be calculated at any given time.

The sections below will go through the main functionality supported by Swim.

2.2.1 Deposit and Redemption

Swim's liquidity pools will support the provision of USDT and USDC on both Ethereum and Solana. Upon deposit to Swim pools on any chain, liquidity providers will be credited with tokens representing their claim on a certain proportion of assets within the pools (LP tokens). Upon redemption, LP tokens are burned and an asset of choice credited to the redeemer's wallet.

Below is the example flow for a user depositing a combination of ERC20 and SPL tokens to receive LP tokens:

1. Connect both Ethereum and Solana wallets to Swim
2. User approves spend for ERC20 USDT and USDC on Metamask and SPL USDT and USDC on Sollet
3. User approves the transactions to deposit both SPL and ERC-20 USDT and USDC
4. SPL USDT and USDC is sent from Sollet to Swim's liquidity pool on Solana
5. ERC20 USDT and USDC is sent from the user's Metamask to the Swim smart contract on Ethereum, then to the Wormhole bridging contract. When the tokens are successfully bridged, the wrapped USDT is locked in Swim's liquidity pool
6. As the liquidity pool receives these assets, SPL LP tokens are minted and credited to Sollet.

The diagrams below show the asset flows for deposits and redemptions, and how Swim uses Wormhole to facilitate any combination of token deposit on any chain, offering a significant degree of flexibility.

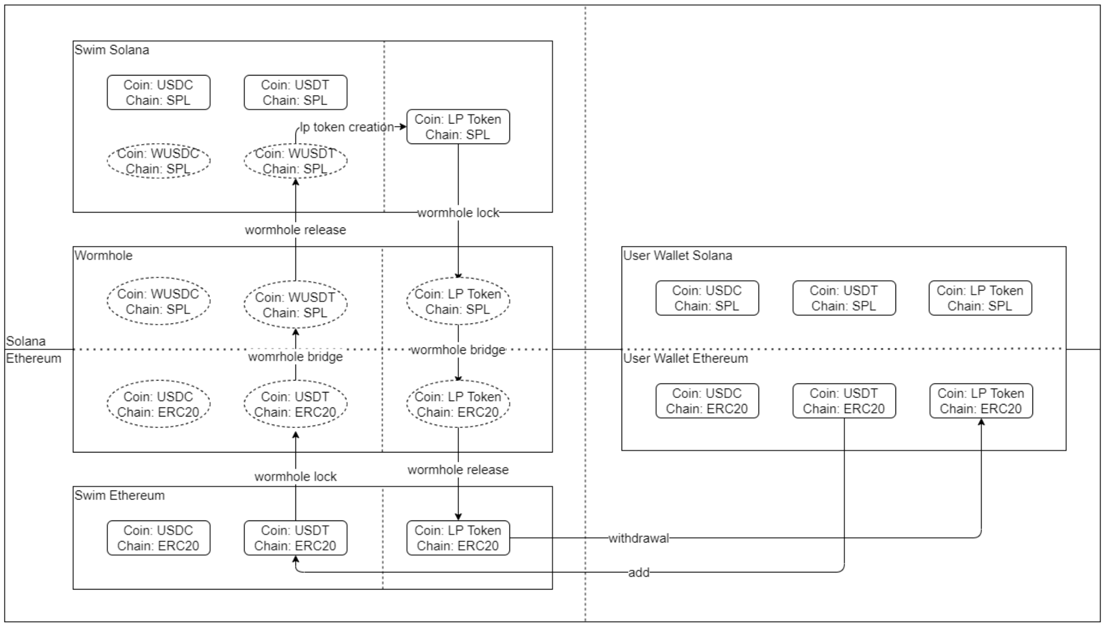

Figure 5: Depositing ERC20 token to receive ERC20 LP token

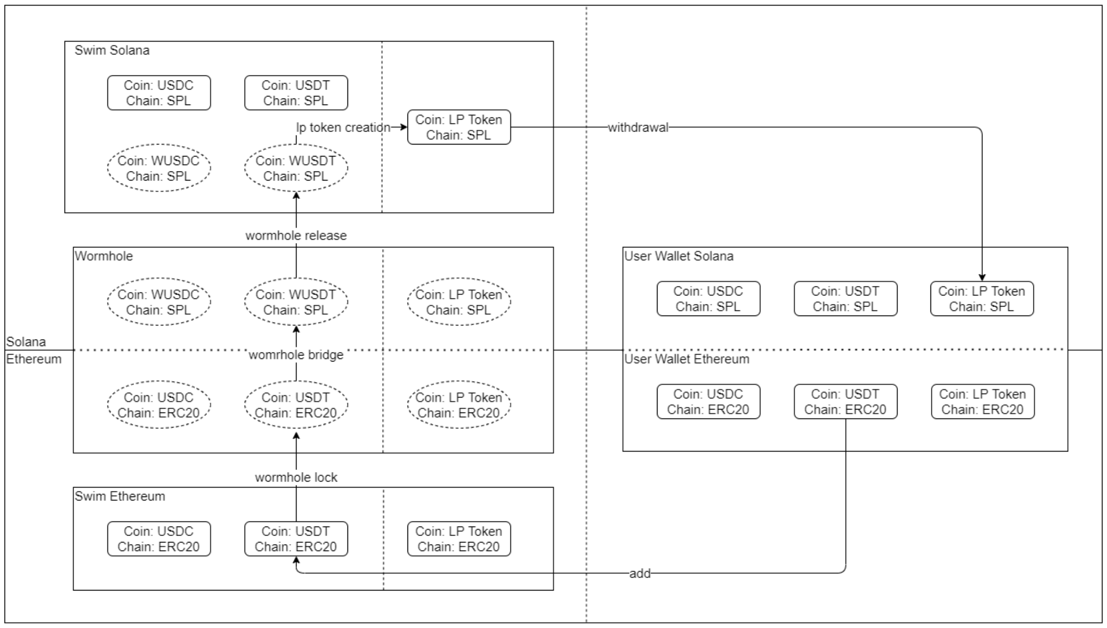

Figure 6: Depositing ERC20 token to receive SPL LP token

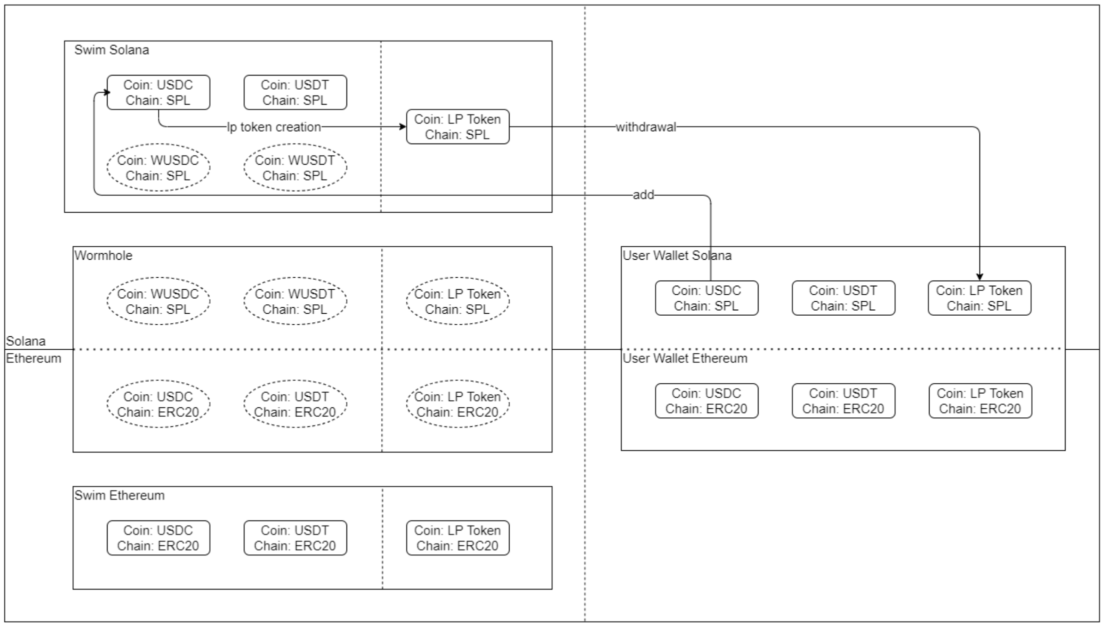

Figure 7: Depositing SPL token to receive SPL LP token

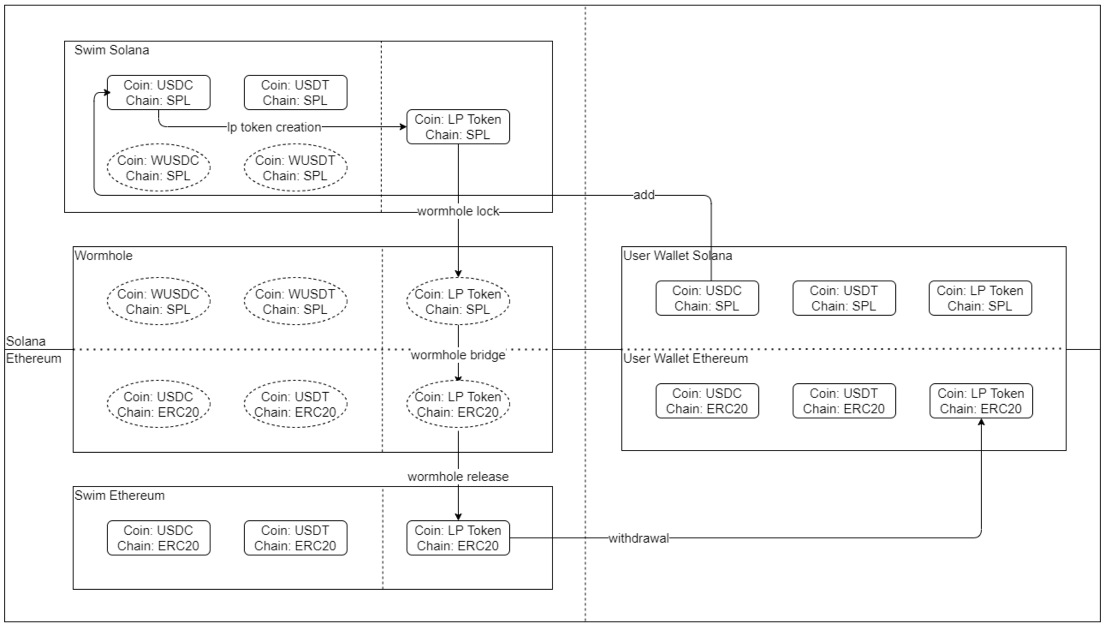

Figure 8: Depositing SPL token to receive ERC20 LP token

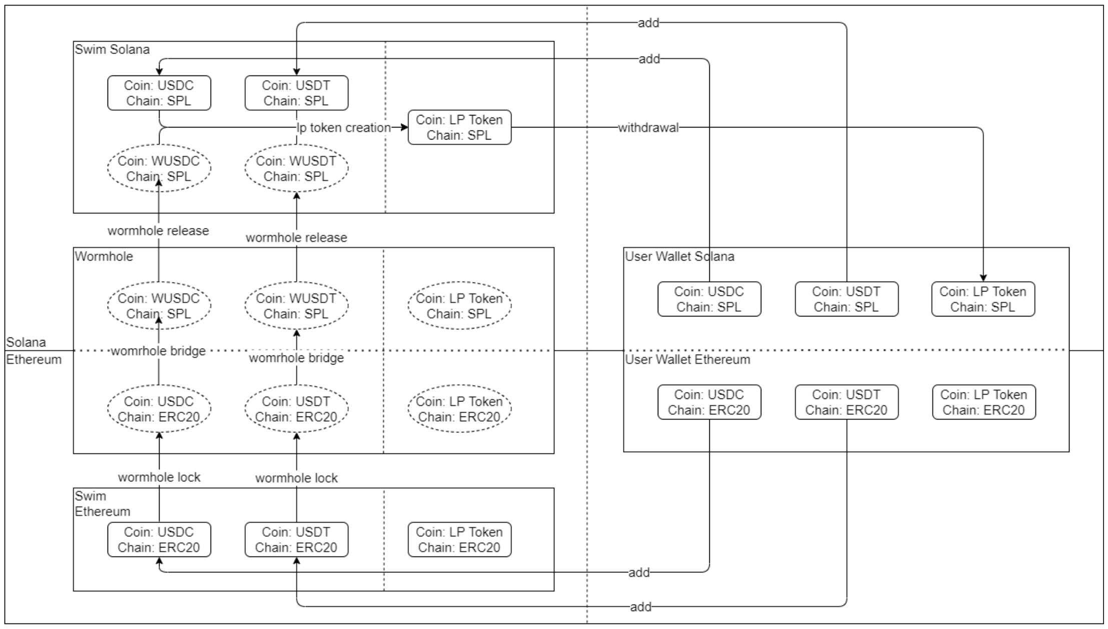

Figure 9: Depositing both ERC20 and SPL token to receive SPL LP token

2.3 Automated Market Maker

With liquidity pools established on both Swim's Solana and Ethereum pools, users will be able to swap between SPL and ERC20 native stablecoins.

2.3.1 Slippage Curve

Swim's AMM is inspired by Curve's proven StableSwap invariant, with necessary modifications to enable cross-chain stablecoin swaps. Like Curve, Swim's slippage curve will be able to support exceedingly low price slippage for tokens that are tokens with relatively similar balances within the liquidity pools.[13]

2.3.2 Generalized AMM

We would like to note that Swim's design is not necessarily limited to stablecoin swaps. With a number of changes to the slippage curve parameters and the introduction of a price oracle, Swim's AMM could support cross-chain trading between assets that have a volatile exchange rate. For example, it would be possible in the future to support SOL/ETH, or even LINK/SRM trading pairs. The main benefit for this is that it allows Swim to leverage the inherently higher liquidity for native assets on each chain, thereby creating on-chain trading pairs that today's DeFi ecosystem cannot easily support.

2.3.3 Fees

Traders will incur a trading fee to the protocol in addition to any slippage (or bonus) incurred (or received) according to pool composition. These fees are necessary to compensate liquidity providers for the opportunity cost of providing their assets and have been calibrated at inception to be competitive with similar AMMs available. The fee parameter will be adjustable and can be raised or lowered as needed.

2.4 AMM Functions

Swim's AMM supports: cross-chain swaps (i.e., Ethereum to Solana), native chain swaps (i.e., Solana to Solana). Eventually, swaps between non-Solana blockchains (i.e., Ethereum to Binance Smart Chain) will also be possible. This section details the mechanisms in more detail.

2.4.1 Cross-Chain Swaps

Below is an example for a ERC20 to SPL swap. The same process applies in reverse for SPL to ERC20 swaps as can be seen from the diagrams further below.

1. Trader connects both Ethereum and Solana wallets to Swim
2. A transaction is submitted on Ethereum for a ERC20 (on Ethereum) to SPL (on Solana) swap
3. The specified ERC20 swap token is sent from the user's ETH wallet to the Swim (Ethereum) liquidity pool
4. Swim smart contracts detect that the respective ERC20 token has been received in the liquidity pool
5. The Swim smart contract on Ethereum then sends the received tokens via Wormhole to Swim's smart contracts on Solana
6. The wrapped ERC20 token arrives at Swim's smart contract on Solana, which then determines the correct bonus/slippage and execution price based on the algorithm
7. Swim credits user's SOL wallet with the native SPL token specified (from the Swim Solana liquidity pool) based on the bonus/slippage curve. They have now successfully swapped from native ERC20 on Ethereum to native SPL on Solana.

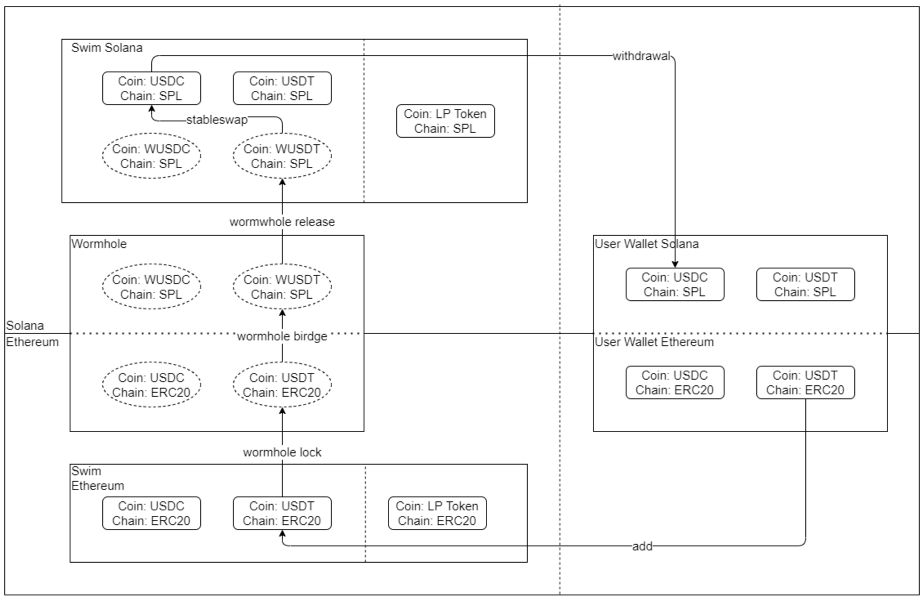

Figure 10: ERC20 to SPL token swap

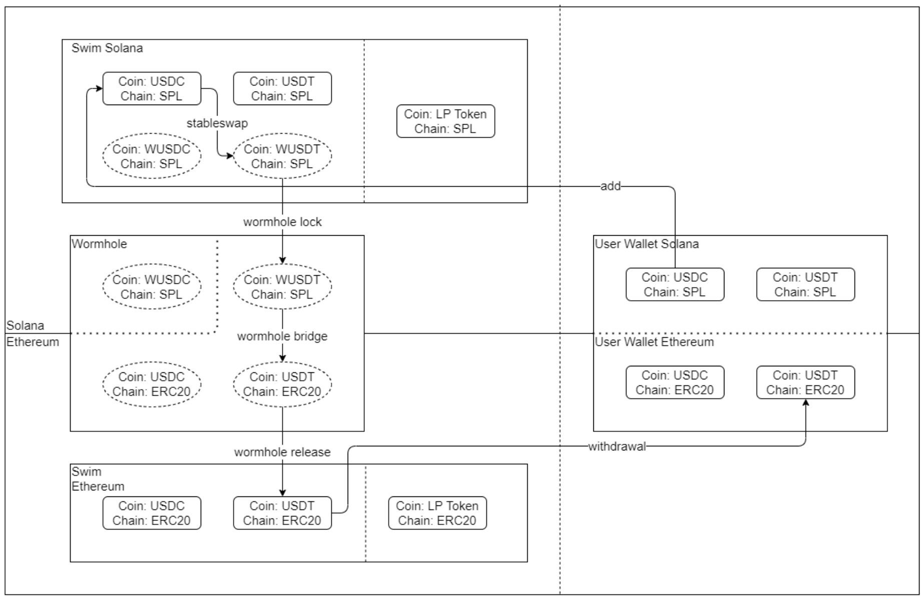

Figure 11: SPL to ERC20 token swap

2.4.2 Native Chain Swaps

The steps and diagram below detail an ERC20 to ERC20 swap, which will involve more complexity and steps due to computations being done on Solana.

1. Trader connects both Ethereum and Solana wallets to Swim
2. A transaction is submitted on Ethereum for a native ERC20 to ERC20 swap (on Ethereum).
3. The specified ERC20 swap token is sent from the user's ETH wallet to Swim's Ethereum liquidity pool.
4. The tokens are bridged via Wormhole to Solana, where they become SPL tokens as a wrapped representation of the ERC20.
5. The wrapped ERC20 token arrives at Swim's smart contract on Solana, which then determines the correct bonus/slippage and execution price based on the algorithm.
6. The Swim smart contract on Solana then releases the other wrapped ERC20 token back via Wormhole to the Swim smart contract on Ethereum based on the amount calculated via the slippage curve.
7. Swim smart contract receives the requested ERC20 token in the Swim Ethereum liquidity pool and credits the user's ERC20 wallet with the native ERC20 token specified from the Swim Ethereum liquidity pool.

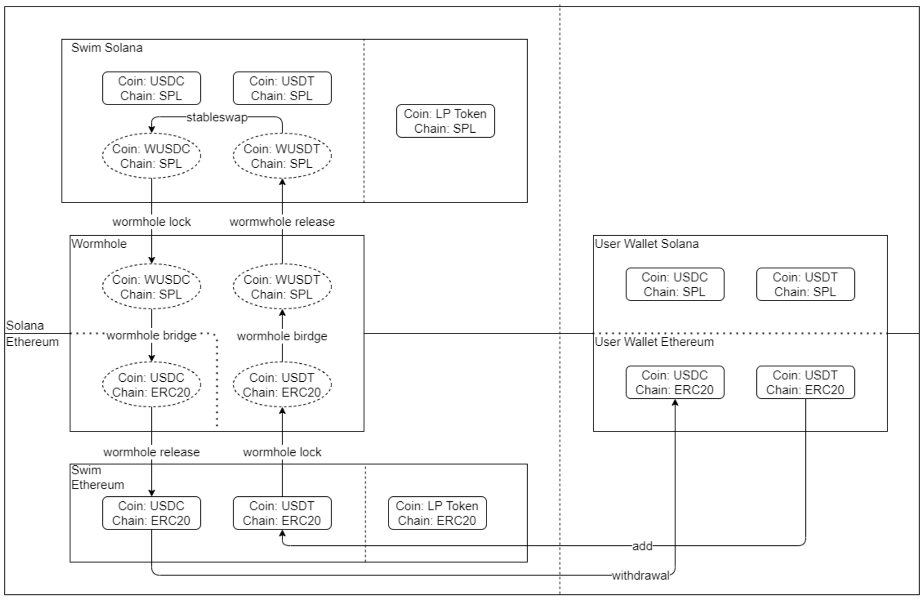

Figure 12: ERC20 to ERC20 token swap

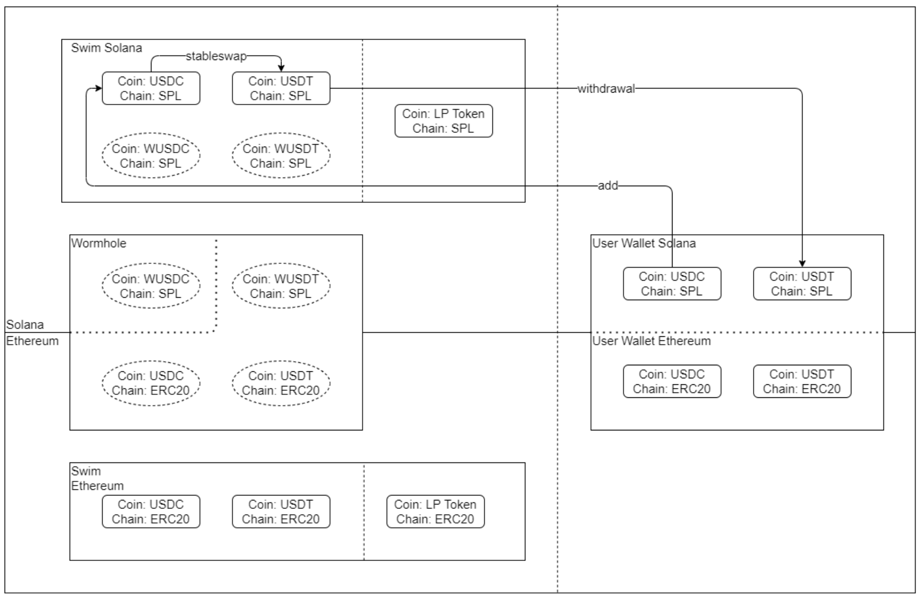

Figure 13: SPL to SPL token swap

3 Tokenomics

There will be a maximum supply of 1,000,000,000 SWIM tokens, broken down as per the table below:

Description	**Details**	**Percentage**
Liquidity mining	Target emission rate of 10% in the first year	30%
Partnership and ecosystem incentives	Pending future partnerships and ecosystem incentive programs	35%
IEO liquidity provision	Unlocked at TGE	10%
Team	Locked for 1 year, linear vesting for 36 months thereafter	20%
Seed round	Locked for 1 year, linear vesting for 36 months thereafter	5%

4 Governance

The current implementation plan is to have the authority to modify Swim's contracts initially be held by the Swim team subject to a time lock. A time lock allows sufficient time for the community to review and discuss any changes. The Swim team plans to transition towards a DAO governance model in due time, where SWIM token holders will directly vote on and enact modifications.

5 Risks

The Swim Protocol will only be as secure as its least secure component. Any vulnerabilities in Wormhole would pose a systemic risk for participants of the system. While traders using only the swap functions will only be exposed temporarily to Wormhole-specific risks, liquidity providers would be exposed constantly to this risk. These risks may not be mitigated even as Wormhole v2 reduces the need for assets to be locked within Wormhole.

Although stableswap AMMs have so far proven to be a relatively robust model, liquidity providers remain exposed to the risk of all assets in their chosen liquidity pools. In the unlikely scenario that one of the assets in the liquidity pool suffers a significant loss in value, liquidity providers will race to withdraw unaffected assets, leaving the less valuable asset in the pools. The last liquidity providers to withdraw their assets may incur substantial losses.

6 Roadmap

This section details the roadmap for the Swim Protocol.

2021 Q3	Public announcement Fundraising Alpha testing and devnet
2021 Q4	Mainnet launch SWIM token genesis event Further integrations and collaboration with Solana projects
2022 Q1	Other network support (BSC and others) Generic asset pairs (i.e., ETH/SOL) DAO launch
2022 Q2	Cross-chain project collaboration

7 References

[1] DeFi Llama. 2020. URL `https://defillama.com/`.

[2] Rekt DAO. Anyswap - REKT. 2021. URL `https://rekt.news/anyswap-rekt/`.

[3] Rekt DAO. Chainswap - REKT. 2021. URL `https://rekt.news/chainswap-rekt/`.

[4] Rekt DAO. Poly Network - REKT. 2021. URL `https://rekt.news/polynetwork-rekt/`.

[5] Rekt DAO. THORchain - REKT 2. 2021. URL `https://rekt.news/thorchain-rekt2/`.

[6] Polygon. Plasma Bridge. URL `https://docs.matic.network/docs/develop/ethereum-matic/plasma/getting-started/`.

[7] Paul Gangemi. THORchain. 2021. URL `https://thorchain.org/`.

[8] Dejun Qian. Anyswap. 2021. URL `https://anyswap.exchange/`.

[9] Dan Robinson. HTLCs Considered Harmful. 2019. URL `http://diyhpl.us/wiki/transcripts/stanford-blockchain-conference/2019/htlcs-considered-harmful/`.

[10] Robin Hanson. Logarithmic Market Scoring Rules for Modular Combinatorial Information Aggregation. 2002. URL `http://mason.gmu.edu/~rhanson/mktscore.pdf`.

[11] Hayden Adams, Noah Zinsmeister, and Dan Robinson. Uniswap v2 Core. 2020. URL `https://uniswap.org/whitepaper.pdf`.

[12] Leopold Schabel. Introducing the Wormhole Bridge. 2020. URL `https://medium.com/certus-one/introducing-the-wormhole-bridge-24911b7335f7`.

[13] Michael Egorov. StableSwap - efficient mechanism for Stablecoin liquidity. 2019. URL `https://curve.fi/files/stableswap-paper.pdf`.

8 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. This paper reflects current opinions of the authors and is not made on behalf of the Swim team or their affiliates and does not necessarily reflect the opinions of Swim team or their affiliates or individuals associated with them. The opinions reflected herein are subject to change without being updated.