zkLink - A Multi-Chain Rollup Infrastructure Based On Zero-Knowledge Technology

dev@zklink.org

Abstract

The lack of standardization and interoperability among blockchains remains a significant challenge for crypto mass adoption. These challenges include a lack of asset pricing capabilities on decentralized exchanges (DEXs), caused by liquidity fragmentation and increased trading costs for multichain assets, which has decayed the user experience, and increased complexity in deploying dApps across chains due to different programming languages and tools. As a result, the absence of user-friendly DeFi applications has compelled traders to resort to using centralized exchanges (CEXs) for trading and asset management. However, this requires users to relinquish custody of their funds to a CEX, which is in contradiction to the blockchain's principles of decentralization and self-sovereignty.

zkLink tackles these challenges by building a decentralized multi-chain ZK-Rollup infrastructure that connects the various L1 and L2 ecosystems, aggregating native crypto assets from different blockchain ecosystems into a unified interface. zkLink simplifies the complexity of multi-chain dApp deployment, unifies fragmented liquidity across ecosystems and significantly reduce the trading cost, thereby returning pricing power back to DEXs.

1 Background

As the blockchain space undergoes rapid advancement, there is an increasing number of layer 1 blockchains such as Ethereum, Solana, Avalanche, and Ethereum layer 2 scaling solutions including ZK-Rollups and Optimistic Rollups. Therefore, an intricate multi-chain, multi-layer landscape has emerged as the current reality and is deemed the inescapable future of the crypto ecosystem where users navigate between the different L1 chains and L2 rollups to satisfy their particular requirements, utilizing a diverse range of crypto tokens.

Despite this multi-chain, multi-layer landscape creating considerable value for crypto users worldwide, it has also brought about unforeseen challenges. These challenges include liquidity silos that are isolated to specific chains, increased trading costs for multichain assets, fragmented user experience, and a complex application development environment.

1. **Liquidity Fragmentation**

   The rise of various new blockchain and rollup networks has led to fragmented liquidity. As a result, this situation of liquidity fragmentation makes it difficult for users to manage their assets and carry out transactions across chains, resulting in lower capital use rates.

2. **Multi-chain Product Deployment Challenge**

   As the Ethereum Layer 2 ecosystems and alternative Layer 1 chains continue to grow, they form liquidity silos. Developers, therefore, need to deploy their products on various networks in order to attract users and liquidity. However, different programming languages and tools, such as EVM, CairoVM, and Solana VM, present notable challenges to developers. Even entirely compatible protocols like multiple EVM compatible chains (Rollups) have many subtle differences.

3. **Navigation Complexity and High Cost**

   Previously, users found it difficult and expensive to navigate between blockchains, for example, to swap tokenA on chainA for tokenB on chainB through a DEX. This procedure turned out to be quite intricate and involves multiple fees.

- First, it requires the installation of a wallet and purchase of the gas token for chainB.
- Next, users need to trade tokenA for a stablecoin or another intermediary token which can be bridged to chainB.
- Then, users need to purchase tokenB on a local DEX.

The emergence of numerous cross-chain asset bridge applications has improved this experience to an extent. However, the cost of cross-chain token exchange remains high, and users still struggle with conducting affordable and seamless token interactions across different chains.

4. **Security Risks During Inter-chain Transactions**

   Preserving asset security poses a formidable technical challenge in cross-chain transactions. For example, in recent years, cross-chain asset bridges have been one of the most susceptible components to hacks in the crypto ecosystem.

2 Introduction

zkLink addresses the above challenges of blockchain interoperability and standardization by building a multi-chain rollup infra protocol to simplify multi-chain dApp deployment, and resolve liquidity fragmentation. It leverages zero-knowledge proof technology to provide a high throughput, low-cost App Rollup deployment solution.

By harnessing the potential of zero-knowledge proof technology, the zkLink protocol features key functionalities such as:

- Multi-chain liquidity aggregation across L1 blockchains and L2 rollups.
- Quick multi-chain product deployment with SDK and APIs.
- A trading-specific-zkVM, empowering high-throughput, low-cost App Rollup solution for high performance financial applications such as Order Book DEX.

3 zkLink Protocol Design

3.1 Protocol Overview

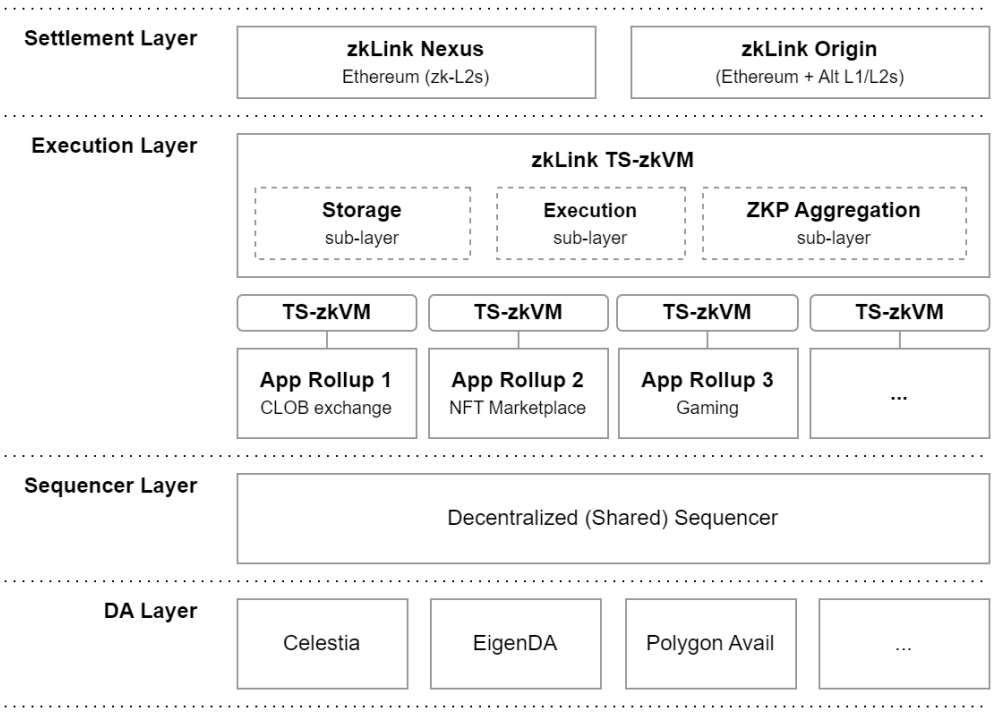

Figure 1: zkLink Protocol Architecture Overview

As shown in Figure 1 above, zkLink Protocol is a multi-chain ZK-Rollup Infrastructure composed of four layers: the settlement layer, the execution layer, the sequencing layer, and the DA layer. These four layers are decoupled for customizable App Rollup deployment.

The following sections will introduce the protocol from the perspective of the four layers. However, it is important to note that the core value proposition and most innovations of the zkLink protocol are related to the execution layer solutions and the settlement layer solutions.

3.2 Settlement Layer

A classic ZK-Rollup network typically selects a single chain, i.e, Ethereum as the settlement layer to verify the proofs and settles the transactions. The settlement layer maintains the security and integrity of off-chain transactions.

In order to aggregate liquidity and native assets across L1s and L2s, zkLink introduces two new paradigms of settlement layer deployment solutions to its rollup infra framework - zkLink Nexus and zkLink Origin.

3.2.1 Working Principal of A Multi-Chain ZK-Rollup

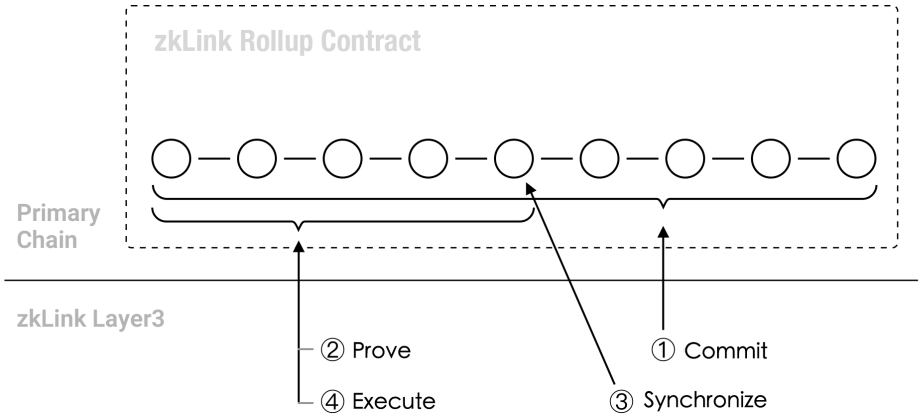

Figure 2: working principal of a multi-chain ZK-Rollup

In comparison to a classic ZK-Rollup architecture, zkLink proposes a new type of ZK-Rollup architecture which requires to deploy the rollup contracts on multiple L1 blockchains and L2 networks at the same time, thus is called a multi-chain ZK-Rollup.

In the architecture of a multi-chain ZK-Rollup, in order to optimize the verification cost, only one chain among the connected chains will be designated as the **primary chain**, which is responsible for ZKP verification, while the other chains will act as **secondary chains** that do not need to execute ZKP verification. It is mathematically equivalent to completing the verifications on all chains.

The working principal of a classic ZK-Rollup typically contains three stages: the Commit stage, Prove stage, and Execute stage. zkLink multi-chain ZK-Rollup architecture adds a Sychronize stage after the Prove stage.

The 4 stages are briefly described as below:

1. **Commit:** The sequencer submits all rollup data (excluding actual transaction data in Validium mode) to the primary chain, where the contract on the primary chain executes the complete zero knowledge proof verification process. Additionally, the sequencer will only commit block information to a specific secondary chain when a particular layer3 block contains transactions related to that secondary chain.
2. **Prove:** ZKPs are submitted to the primary chain and verified by the smart contract. This process occurs only on the primary chain. Therefore, the primary chain is required to have the capability to verify zk-SNARKs proofs.
3. **Synchronize:** The purpose of the Synchronize stage is to achieve states synchronization across the connected chains. After completing ZKP verification, the primary chain calculates the sync hashes for the different secondary chains. The primary chain, during the commit stage, has received transaction information related to all secondary chains, allowing it to calculate the sync hashes of the secondary chains.

   And then, the secondary chains use the cross-chain messaging mechanism, which is detailed in section 4.2, to send their respective sync hashes to the primary chain. Once the primary chain receives all the sync hashes from the secondary chains, it compares them with its own calculated sync hash. If they are consistent, the primary chain notifies the secondary chains that they can proceed with transaction settlement and asset withdrawal.
4. **Execute:** This step aligns with the typical process found in a classic ZK-Rollup, which involves approving requests for fund withdrawals.

3.2.2 Nexus: Settlement on ETH ZK-L2s

A zkLink Nexus L3 rollup settles the transactions and transition of states directly on Ethereum's ZK-L2s, and the L2s settle on Ethereum. Hence the finality of the transactions on a zkLink Nexus L3

rollup will be ultimately achieved on Ethereum layer1.

Figure 3: Security Inheritance of Nexus Layer 3

The settlement process of a Nexus L3 rollup is shown in figure 3. Nexus rollup posts data and ZKP to the connected ZK-L2s, where the correctness of the states and transactions is verified via the verifier contract. Meanwhile, the ZK-L2s also post transactions and states to Ethereum with ZKP verified. In essence, from Nexus L3 to the ZK-L2s, and then to Ethereum, each step is verifiable with ZKP, ensuring the correctness of the L3 off-chain states. Therefore, it could be stated that a zkLink Nexus L3 rollup inherits the security of Ethereum.

3.2.3 Orgin: Settlement on ETH and Alt-L1s

In contrast to Nexus, zkLink Origin's architecture allows for the integration with OP-L2s and Alt-L1s, in addition to ZK-L2s.

A zkLink Origin rollup will settle the transactions and transition of states on Ethereum, ALt-L1s, OP-L2s, and ZK-L2s, corresponding to the actual networks connected, as long as one of the networks supports zk-SNARKs proof verification, and there is a secure and fast way to achieve states synchronization across the networks.

To establish a fast and secure communication mechanism between the alt-L1s and Ethereum, zkLink introduces a Light Oracle Network for cross-chain message transfer, the process of which is described in section 4.3. Therefore, the security assumption of zkLink Origin is that the rollup sequencers, responsible for packaging transactions, cannot collude with all nodes of the Light Oracle Network for malicious activities.

Security Assumptions

Table 1: Security Assumptions of zkLink Nexus and Origin

	Solutions	**Security Assumption**
Classic zkRollup	zkSync Era, StarkNet, Scroll, Polygon zkEVM, Linea, Taiko...	Ethereum Equivalent
Multi-chain zkRollup	zkLink Nexus	Ethereum Equivalent
	zkLink Origin	The Sequencer and the Light Oracle Network cannot collude for malicious activities.

Supported Networks

Table 2: Supported Networks of zkLink Nexus and Origin

		zkLink Nexus	zkLink Origin
Mainnet	Ethereum	✓	✓
Ethereum zkRollup	zkSync Era	✓	✓
	StarkNet	✓	✓
	Scroll	✓	✓
	Linea	✓	✓
	Taiko	✓	✓
	Polygon zkEVM	✓	✓
	...		
Ethereum OpRollup	Arbitrum One		✓
	Optimism		✓
	...		
EVM-compatible Alt L1s	BNB Chain		✓
	Polygon POS		✓
	Avalanche		✓
	...		

3.3 Execution Layer

Execution entails executing transactions that update the state correctly. Thus, execution must ensure that only valid transactions are executed, i.e., transactions that result in valid state machine transitions.

There are two types of execution environments of zkLink multi-chain rollup infra solution: the TS-zkVM and the zkEVM. zkLink App Rollup developers will be able to choose from the two options based on their specific needs.

3.3.1 TS-zkVM for App Rollup

zkLink Trading-Specific-zkVM (TS-zkVM) is based on tailored core ZK circuits and Risc0 zkVM extension ZK circuits, of which the details are specified in section 4.1. The TS-zkVM offers high-throughput and low-cost execution for App Rollup developers. It supports various order book product features, including but not limited to spot trading, derivatives trading, NFT trading, etc.

Different developers and applications may have varying requirements for cost and security, zkLink App Rollup developers are able to compose the key components including DA, Sequencer, and Settlement layer.

3.4 Sequencing Layer

The sequencing layer is a pivotal component in rollup systems, primarily responsible for receiving user transactions, sequencing the transactions and bundling them into batches. These batches are then committed to the settlement layer. Additionally, in scenarios where the system employs an external DA layer, the sequencer also ensures efficient transmission of transaction data to the DA layer.

Similar to other rollups, App Rollups will start with a centralized sequencer model. While this approach offered certain development efficiencies, it also presentes challenges and risks, such as potential single point of failure, transaction censorship and issues around miner extractable value (MEV), affecting network fairness and transparency.

To address these concerns, zkLink protocol aims to incorporate decentralized sequencer solutions. These solutions, including platforms like Espresso, Astria, and Fairblock, aim to mitigate centralization risks by processing and validating transactions across a distributed node network. This strategy will

not only boost network security and transparency but also strive to offer a more secure, equitable, and efficient rollup solution to its users.

3.5 DA Layer

Data Availability (DA) entails making the transaction data available. DA layer is a critical component for rollups to ensure the capability to reconstruct the the rollup states when the rollup service halts unexpectedly.

Both zkLink Nexus and zkLink Origin by default support the primary chain as the DA layer.

Secondly, both Nexus and Origin support Validium, which involves an external DA solution. zkLink will integrate various third party modular DA solutions, such as Celestia, EigenDA, Avail, etc., to meet diverse demands from developers.

Furthermore, zkLink DAO will have the option to organize a Data Availability Committee (DAC). This DAC serves as another option for developers, providing an additional choice for data availability assurance.

4 Key Components and Process Specifications

4.1 Trading Specific zkVM (TS-zkVM)

The TS-zkVM is a high-efficiency ZKP execution environment specifically designed for high-performance financial products such as CLOBs, and it is constructed by zkLink. TS-zkVM extends and integrates with the risc0 zkVM outside of the precompiled core circuit.

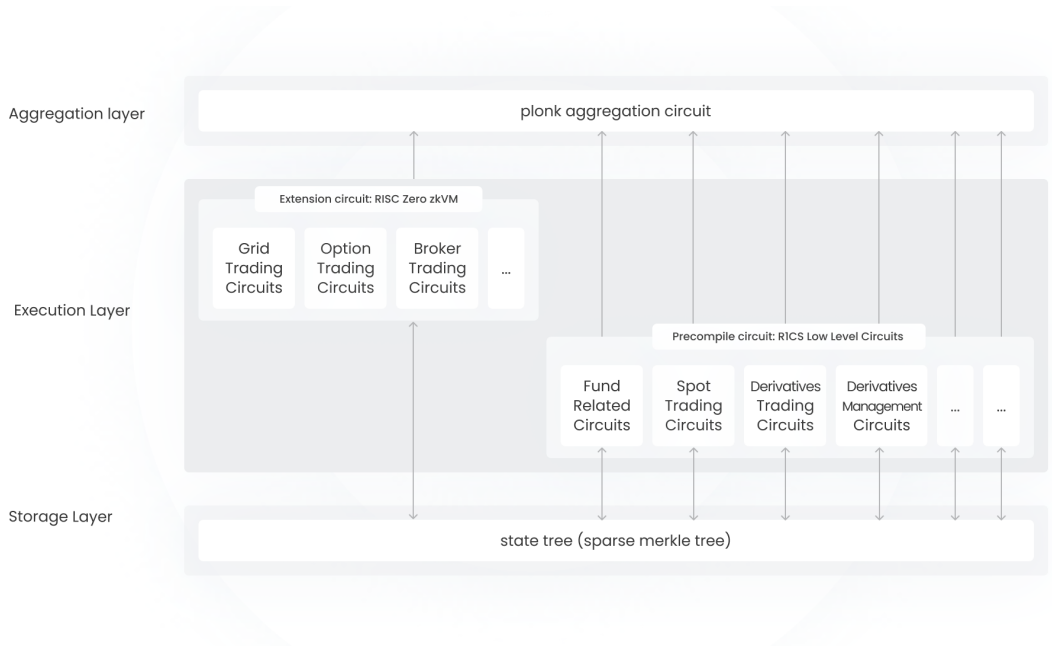

Figure 4: TS-zkVM Architecture

The diagram in Figure 4 above shows a high-level overview of the TS-zkVM architecture, which is divided into three sub-layers.

4.1.1 Storage sub-layer of TS-zkVM

The storage of zkLink protocol utilizes a Sparse Merkle Tree (SMT) to store the state. This data structure provides the system with an efficient and secure way to maintain and verify changes in the state. Compared to the SMT in the EVM, the customized and optimized SMT of zkLink TS-zkVM is more streamlined and better adapted to the needs of high-frequency financial transaction scenarios.

With in-depth customization, the SMT has been specifically tailored to suit particular business needs, thereby achieving exceptional zero-knowledge proof performance.

4.1.2 Execution sub-layer of TS-zkVM

The execution sub-layer of TS-zkVM is divided into two parts, each designed for different performance and scalability requirements:

- **Precompiled circuit:** Requires high performance and includes various transaction-related circuits, such as:
  - Fund Related Circuits (Deposit, Withdrawal, Transfer)
  - Spot Trading Circuits
  - Derivatives Trading Circuits
  - Derivatives Management Circuits (Liquidation, ADL, Funding)
  - Oracle Verification Circuits
  - Authorization Circuits (Passkey Verification, Social Login Verification)

  These circuits are specifically optimized to handle corresponding financial transactions, ensuring efficiency and high response speed when the system processes transactions. Each sub-circuit corresponds to different types of transaction operations or financial instruments and handles the most performance-intensive tasks.
- **Extension circuit** The extension circuit is built using risc0 zkVM on the same Sparse Merkle Tree (SMT), and the proofs generated by the zkVM are subsequently verified by the plonk aggregation circuit in the aggregation layer. Developers can perform custom operations using risc0 zkVM, such as:
  - Grid Trading Circuits
  - Option Trading Circuits
  - Broker Trading Circuits
  - ...

A SDK based on risc0 zkVM will be provided, which includes common operations on the TS-zkVM SMT, such as: verifying a Merkle Tree Proof for a specific account, checking if the balance change of an account complies with certain patterns, etc.

Of course, for common features like Grid Trading and Broker Trading, zkLink will have prepared corresponding circuit templates in advance to enable developers to build quickly.

4.1.3 ZK Proof Aggregation sub-layer of TS-zkVM

The aggregation sub-layer is responsible for aggregating the different transaction proofs and producing a proof to verify the correctness of the entire batch of transactions, which achieves increased efficiency in proof verification and reduced on-chain costs for transactions.

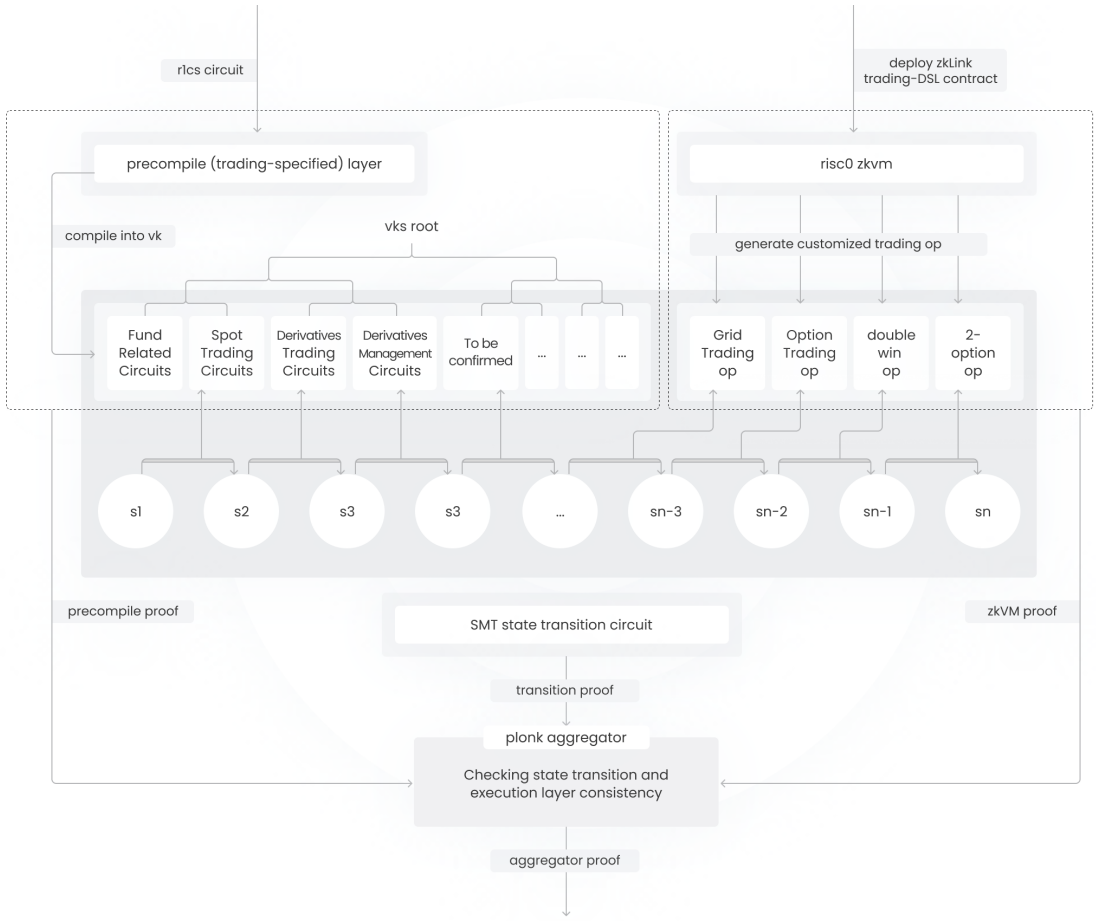

Figure 5: Aggregation Detail of TS-zkVM

Aggregation Details The aggregation sub-layer is composed of the following modules:

- **Precompile Module:** This module includes trading-specified precompiled circuits, which need to be compiled into verification keys (vk). This provides the system with a flexible way to generate customized circuits for specific types of transactions and integrate them into the entire verification system.
- **SMT State Transition Circuit:** The state transition circuit uses an SMT to record states, with changes in the root representing all users' ledger changes, ensuring the correctness and consistency of the system state. Each state transition (S1, S2, S3, ..., Sn) is constrained by the circuit to generate a transition proof.
- **zkVM:** This section demonstrates how to deploy zkLink Trading-DSL contracts using risc0 zkVM to generate custom trading operations (op). This mechanism allows users to customize trading logic and ensures that these logics are correctly executed and verified.
- **PLONK Aggregator:** The PLONK aggregator is responsible for checking the consistency of state transitions and the execution layer. It integrates proofs from different sources and generates an aggregated proof. This process enhances verification efficiency and reduces the costs required for on-chain verification.
- **Proofs:** The architecture includes various types of proofs:
  - **Precompile Proof:** Ensures the correctness of the precompile layer.
  - **Transition Proof:** Ensures the correctness of state transitions.
  - **zkVM Proof:** Ensures the correctness of custom trading operations.

Interactions Between Modules

- Application Specific Circuits are eventually precompiled into verification keys, and these verification keys are stored in the root node of the vk tree.
- zkVM allows for the deployment of Trading-DSL contracts, generating custom trading operations.
- Precompiled circuits, risc0 zkVM, and SMT State Transition Circuits are interconnected, ensuring the correctness of each state change.
- All proofs are collected by the Aggregator and verified in the aggregation circuit — to check if the transition set commitment in the transition proof is included in the (Algorithm id, Input, Output) tuple vector of the precompile proof and zkVM proof, a membership proof verification is also needed in the aggregation circuit. This ultimately produces an Aggregation Proof for on-chain verification.

4.2 Multi-Chain State Synchronization

4.2.1 Deposit fraud prevention: Multi-chain infra security challenge

The main challenge of building a unified trading infra natively deployed across various networks is the risk of **deposit fraud**. When having multiple chains or rollups in the same setup, each network is isolated – they are not able to perceive the states of the others because they can not directly access external information.

When the fund deposit information from the different chains need to converge within zkLink protocol, the potential for deposit fraud emerges. A bad sequencer may falsely inform the primary chain about a fake deposit on one secondary chain, e.g, the sequencer may inform the primary chain about a 100 USDT deposit on a secondary chain which may not exist. In such scenarios, without effective verification mechanisms, it could lead to loss of user funds.

The challenge centers on guaranteeing the states consistency between the primary chain and the secondary chains. Therefore, it could be stated that the issue of deposit fraud is fundamentally a matter of multi-chain state consistency. To address this issue, we need a verification mechanism that is both fast and secure, ensuring all chains stay synchronized, and no one can alter or falsify information.

4.2.2 Nexus solution: multi-chain state synchronization via Ethereum

All networks connected with zkLink Nexus have to be a ZK-Rollup L2 deployed on Ethereum, which ensures a fast communication with Ethereum through the official rollup bridges.

Firstly, zkLink Nexus L3 sequencer will commit state and proof to the primary chain(assuming Linea is the primary chain) in step 1, 2, 3, as is shown in figure 6 below.

Secondly, the primary chain Linea computes the sync hash of Starknet and zkSync and posts it to the Ethereum contract in step 4. In the meantime, the secondary chains, Starknet and zkSync will compute and post their sync hashes representing the deposit history to the Ethereum contract in steps 5 and 6.

Thirdly, the Ethereum contract will compare the sync hash from Linea with the sync hashes from Starknet and zkSync. If they match, confirmation messages will be sent to the contract on the three ZK-L2s through their official Ethereum rollup bridges in steps 7, 8, and 9.

This messaging mechanism allows for a secure and effective synchronization of states cross-L2s, ensuring that deposit fraud issues on the secondary chains are prevented.

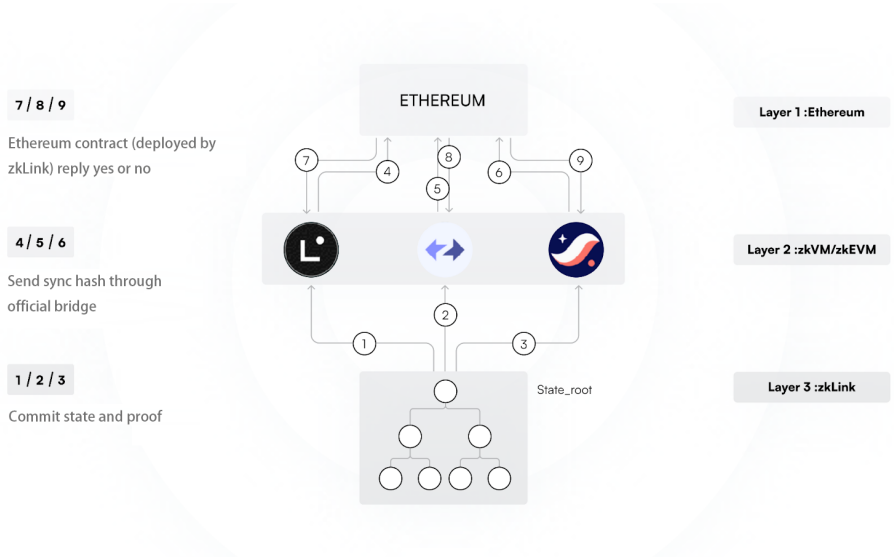

Figure 6: zkLink Nexus Cross-Rollup States Synchronization Process

4.2.3 Origin solution: cross-chain state synchronization via Light Oracle Network

zkLink Origin introduces a light oracle network to facilitate the cross-chain transfer of sync hashes. The light oracle network consists of independant cross-chain general message service providers such as LayerZero, zkBridge, Chainlink, etc., which ensures decentralization of the message transfer process.

As shown in figure 7 below, the zkLink Origin Rollup sequencer will firstly commit state and proof to the primary chain in steps 1, 2, and 3.

Secondly, the secondary chains will post their sync hash to Linea through the light oracle network service in steps 4, 5, and 6.

Thirdly, the Primary contract computes the sync hash of all other secondary chains and compares them with the sync hashes from the secondary chains. If they match, confirmation messages will be sent back to all secondary chains through the light oracle network in steps 7, 8, and 9.

Through the process described above, the cross-chain states are synchronized fast and secure.

In addition, a user that deploys significant capital to a zkLink Orgin Rollup could apply to run a node service of the light oracle network, which will completely eliminate the risk of multi-chain deposit fraud.

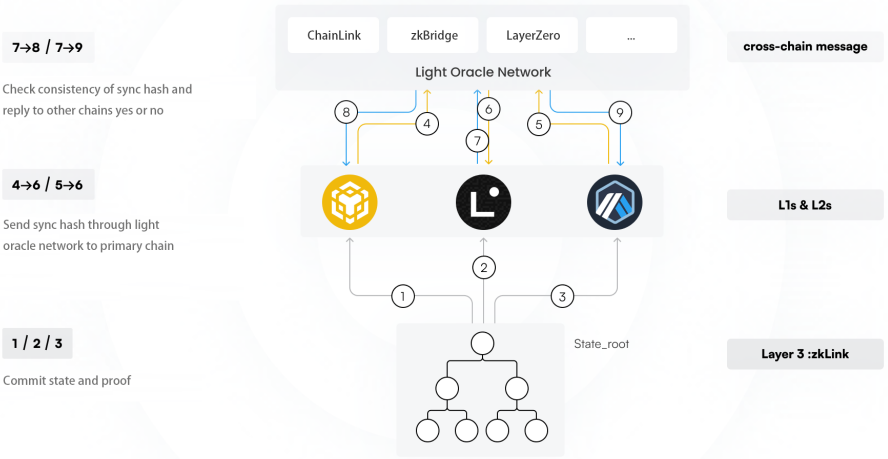

Figure 7: zkLink Origin Cross-Chain State Synchronization Process

5 Key Features of the zkLink Protocol

5.1 Native Asset Aggregation

Applications using zkLink rollup infra solution will be able to access and list the native tokens across the connected L1s and L2s, including FTs and NFTs, allowing users to trade multi-chain assets on a unified user interface. Cross-chain asset bridges are not needed in the process, thus avoiding cross-chain asset bridging risks and bridging fees.

At the same time, multi-chain token portfolios can be managed with a single wallet. For instance, Alice deposits 2 UNI from her Metamask wallet to zkLink on Ethereum, and then deposits 3 BNB from BNB Chain to zkLink from the same wallet address — as a result, Alice will receive 2 UNI + 3 BNB under the same wallet address on the zkLink rollup network. This hypothetical example applies the same to tokens from Polygon, Starknet, zkSync, Linea, Arbitrum, Optimism, Scroll, and Solana, etc. Therefore, users can easily manage their multi-chain token portfolios using a single wallet with a simplified user experience.

5.2 Liquidity Aggregation and Unification

Tokens issued on different L1 chains and L2 rollups by the same entity, for instance, USDT ERC20, USDT BEP20, USDT ARB, etc — will be merged into a single USDT token in the zkLink App Rollups and zkLink L3 network.

The same applies for ETH. As ETH is the native asset for Ethereum and all the Ethereum layer 2 networks, ETH from Ethereum, zkSync, and Starknet, etc., will be merged into a single ETH token, thus eliminating chain disparities.

In summary, tokens of the same kind but issued on various chains will be merged into one single token, fostering unified and aggregated liquidity.

5.3 Customizable App Rollup Deployment

zkLink protocol decouples the four layers of the rollup framework and provides fast and customizable rollup deployment solution. zkLink is focused on the development of execution layer plus settlement layer, and will integrate third party modular solutions for DA layer and sequencing layer, allowing developers to customize the key components to meet diverse demands of different use cases.

- Network Collections And Settlement Layer Solution. Developers can choose which chains the App Rollup can access to, including but not limited to: ETH, BNB Chain, Avalanche, Polygon

PoS, Solana, zkSync, Starknet, Scroll, Polygon zkEVM, Linea, Taiko, Arbitrum, Optimism, Base, etc.

- Execution Environment: TS-zkVM.
- Decentralized Sequencer: Espresso, Astria, Fairblock, etc.
- Modular DA Solutions: In addition to Ethereum, developers can choose Celestia, EigenDA, Polygon Avail, DAC organized by zkLink, etc.

6 Tokenomics

ZKL serves as the native utility token and governance token for zkLink protocol. Utilities wise, ZKL helps developers unlock the access to the zkLink rollup infra service including the proof market. It is also the governance token of zkLink DAO, which governs the development of the protocol.

ZKL will be a standard ERC20 token issued on the Ethereum Mainnet. The total supply of ZKL is capped at 1 billion and is non-inflationary. ZKL will also be made available on the connected chains.

The following diagram illustrates the utilities and economic flow of ZKL.

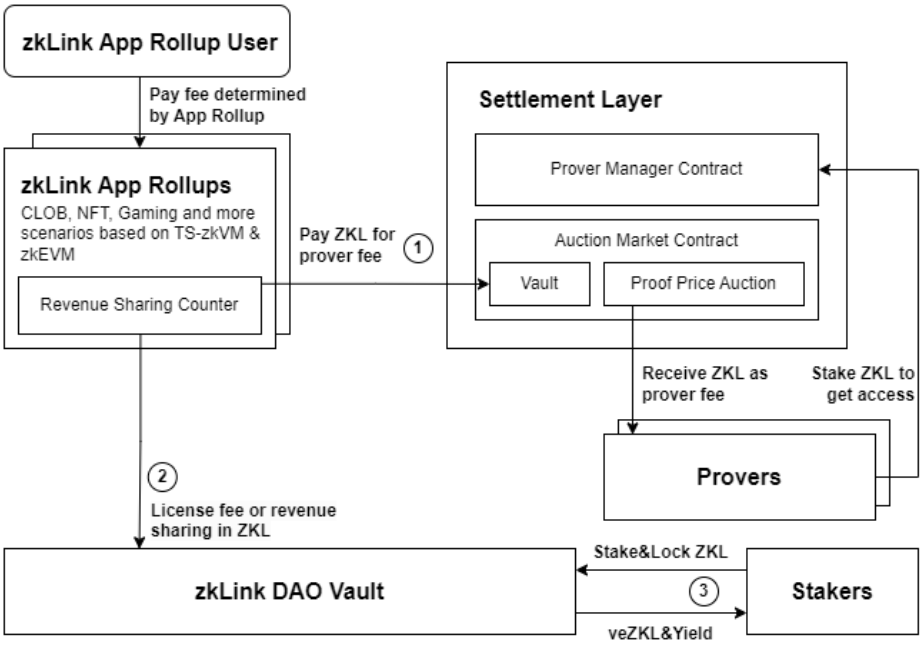

Figure 8: Tokenomics Model

6.1 ZKL as Prover Fee

ZKL is the de facto payment token for the zero-knowledge proof generation service provided by provers. Provers are rewarded with ZKL tokens for their services in generating ZK proofs for the App Rollups.

Provers need to firstly stake a specific amount of ZKL to join the proof market. zkLink DAO will determine the initial staking amount of ZKL needed. Staking is essential to ensure the correctness and timeliness of their operations. The provers that fail to complete a task or exceeding the time requirement will be punished by penalty. The proof price is determined by the proof market auction.

This dynamic mechanism fosters competition and motivates provers to continuously improve their services and contribute to the entire zkLink ecosystem.

6.2 ZKL as Rollup Infra License Fee

App Rollups using zkLink Rollup deployment solution need to pay a license fee priced in ZKL, to the zkLink DAO (the DAO). The DAO will decide on the fee structure for different types of use cases. App Rollups will collect transaction fees or other fees from the users of the App Rollups to sustain the business operation.

To lower the entry barrier, the DAO will provide support for startups using zkLink App Rollup solution. App Rollups can apply to the DAO for a certain amount of ZKL, for covering license fees in the beginning. In this case, the App Rollup will have to agree to share a portion of its future revenue with the DAO. The specific support and revenue sharing model will be coordinated by a vote within the DAO case by case.

6.3 ZKL as Governance token

ZKL is the governance token of zkLink protocol, holders will be able to stake ZKL, get veZKL, initiate proposals, and vote on the proposals to govern the development of the protocol.

In further detail, ZKL holders obtain veZKL by staking ZKL to the governance contract. The amount of veZKL obtained varies depending on the different staking lock-up periods. Governance rights involve cooperation with App Rollups, the management of zkLink DAO vault, and the technical development of zkLink protocol, etc.

7 Conclusion

This paper presents the conceptual design and system structure of zkLink protocol, a decentralized multi-chain rollup infrastructure protocol based on zero knowledge technology.

zkLink addresses the problems of liquidity fragmentation, lack of interoperability, high trading cost, and complexity of multi-chain deployment via a novel multi-chain rollup framework using zk-SNARKs. The fast App Rollup deployment solution based on SDK and APIs enables developers to build products with aggregation of native assets and liquidity across L1s and Ethereum L2s, delivering a seamless multi-chain user experience. And the TS-zkVM provides high-efficiency execution environment specifically designed for high performance and low-cost financial products such as CLOB DEX, NFT trading etc., contributing to the mass adoption of self-custodial, user-friendly DeFi products.

To sum up, zkLink protocol is committed to empowering the next generation of DeFi applications and onboarding the next billion users on-chain via its innovative multi-chain ZK-Rollup infra solution.

8 References

https://docs.celestia.org/developers/build-modular

https://docs.espressosys.com/sequencer/espresso-sequencer-architecture/readme

https://docs.polygon.technology/cdk/

https://layerzero.network/publications/LayerZero_Whitepaper_V2.0.pdf

https://dev.risczero.com/api/zkvm/developer-guide/zkvm-specification

https://l2beat.com/scaling/summary

https://era.zksync.io/docs/dev/