Whitepaper

Abstract

Blockchain scalability has been a long-standing challenge in the world of decentralized finance (DeFi) and decentralized applications (dApps). Sliceledger presents a promising solution to this problem. This whitepaper explores the concept, architecture, and benefits of Sliceledger, aiming to provide a comprehensive understanding of this innovative scaling solution.

Table of Contents

1. Introduction

Blockchain technology has revolutionized various industries by providing trustless, decentralized, and immutable ledgers. However, as Blockchain adoption has grown, so too have concerns about its scalability. To address this, various Layer 2 scaling solutions have emerged, with Sliceledger being one of the most promising and widely discussed.

This whitepaper provides an in-depth analysis of Sliceledger, focusing on their architecture, working principles, benefits, challenges, and potential use cases.

2. Background

2.1. Blockchain Scalability Problem

The fundamental challenge facing Blockchain like Bitcoin and Ethereum is scalability. These networks struggle to handle a large number of transactions efficiently due to their limited processing capabilities. As a result, transaction fees rise, and confirmation time's increase, negatively impact user experience.

2.2. Layer 2 Scaling Solutions

Layer 2 scaling solutions aim to mitigate the scalability issues of Layer 1 Blockchain. These solutions create additional layers on top of the main Blockchain to process transactions more quickly and cost-effectively. Sliceledger is a prominent Layer 2 solution.

3. What is Sliceledger?

3.1. Optimistic Execution

Sliceledger operates on the principle of "optimistic execution," which means that transactions are initially assumed to be valid without immediate validation. This approach contrasts with "pessimistic execution," where every transaction is rigorously checked before execution.

3.2. Sliceledger Concept

The Sliceledger concept involves offloading most transaction processing and data storage from the main Blockchain to a secondary layer. This secondary layer, known as the "Sliceledger chain," aggregates and verifies transactions before submitting a summary to the main Blockchain.

3.3. Key Components

Sliceledger consist of several key components:

- Sliceledger Chain: A Layer 2 chain where transactions are processed and validated.

- Main Chain: The primary Blockchain network where Sliceledger chains submit their summarized data.

- Smart Contracts: These are deployed on the Sliceledger chain and interact with users and applications.

- Bridge: A mechanism for transferring assets and data between the Sliceledger chain and the main chain.

4. How Sliceledger Work

4.1. Transaction Submission

This system depends on two different kinds of transactions: L2 transactions, which take place between two addresses on the L2 chain, and cross chain transactions, which happen between the L1 and L2 chains. The L2 transaction process is explained in a little more detail in the workflow that follows than in the basic workflow.
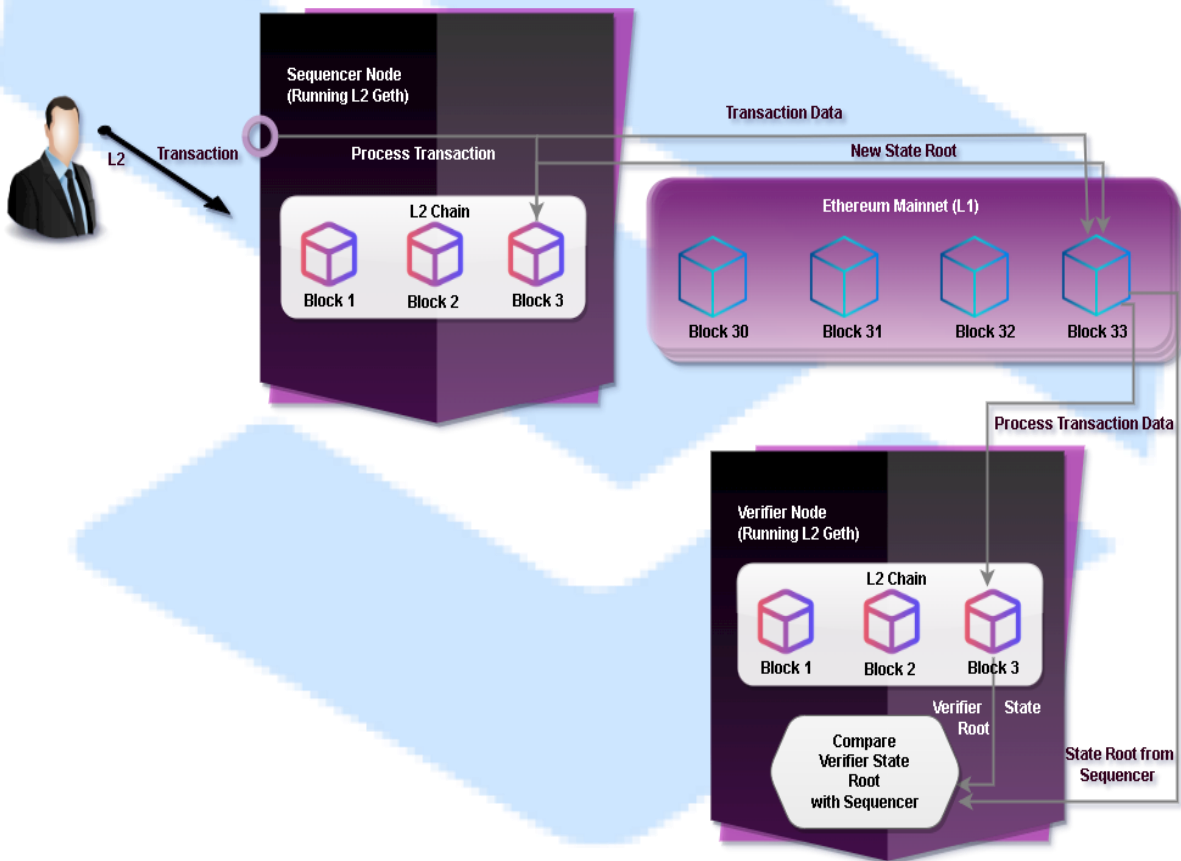
L2 Transaction Workflow

If a transaction is sent to the sequencer node by a user and is found to be legitimate, the sequencer will immediately add it to the L2 chain (note that at this time, just the sequencer node has added this transaction to its copy of the L2 chain). L2 only allows for single-transaction blocks, therefore as soon as a new transaction is added, a new block is added to the chain. Since the sequencer now performs the duty of the miner, there are no miners competing to mine new blocks in L2.

The sequencer will then call a smart contract on L1 (deployed by the Optimism team prior to release) after adding a few transactions to the L2 chain and send it the transaction data for all of those L2 transactions as well as the new state roots of the L2 chain after applying each transaction.

The smart contract on L1 will efficiently (using Sliceledger) store the transaction data and state roots in another smart contract created for storage.

The verifier nodes will include the transaction in their copy of the L2 chain once the transaction data has been deposited on L1.

The user who is being restricted by the sequencer will still be able to input the transaction data and invoke the smart contract themselves. The sequencer will then be required to complete that transaction in a specific amount of time. If not, their link might be reduced.

Furthermore, it has already been noted that the verifier examines the transactions that the sequencer posts to L1. Verifiers do have the option to sync from L2, in which case they would receive new

transactions straight from the sequencer, possibly before they were pushed to L1. The sequencer may post this transaction to L1 but there is no assurance that it will do so with this sync from L2 technique.

Cross Chain Transactions

In order for users to be able to invoke contracts on other chains or transmit ETH/tokens from one chain to another, cross-chain transactions are required in this system. Given that they involve both chains, these transactions have a slightly different workflow than L2 transactions.

L1-> L2 Transactions

The sequencer simply relays the message to the L2 chain in transactions from L1 to L2, which are quite quick. Users will provide the necessary information for their transactions to a bridge smart contract on Layer 1 (L1), and this smart contract will add the transaction to a queue of transactions that the sequencer must add to Layer 2 within a predetermined amount of time. Therefore, the transaction will finally be transmitted to the L2 chain by the sequencer.
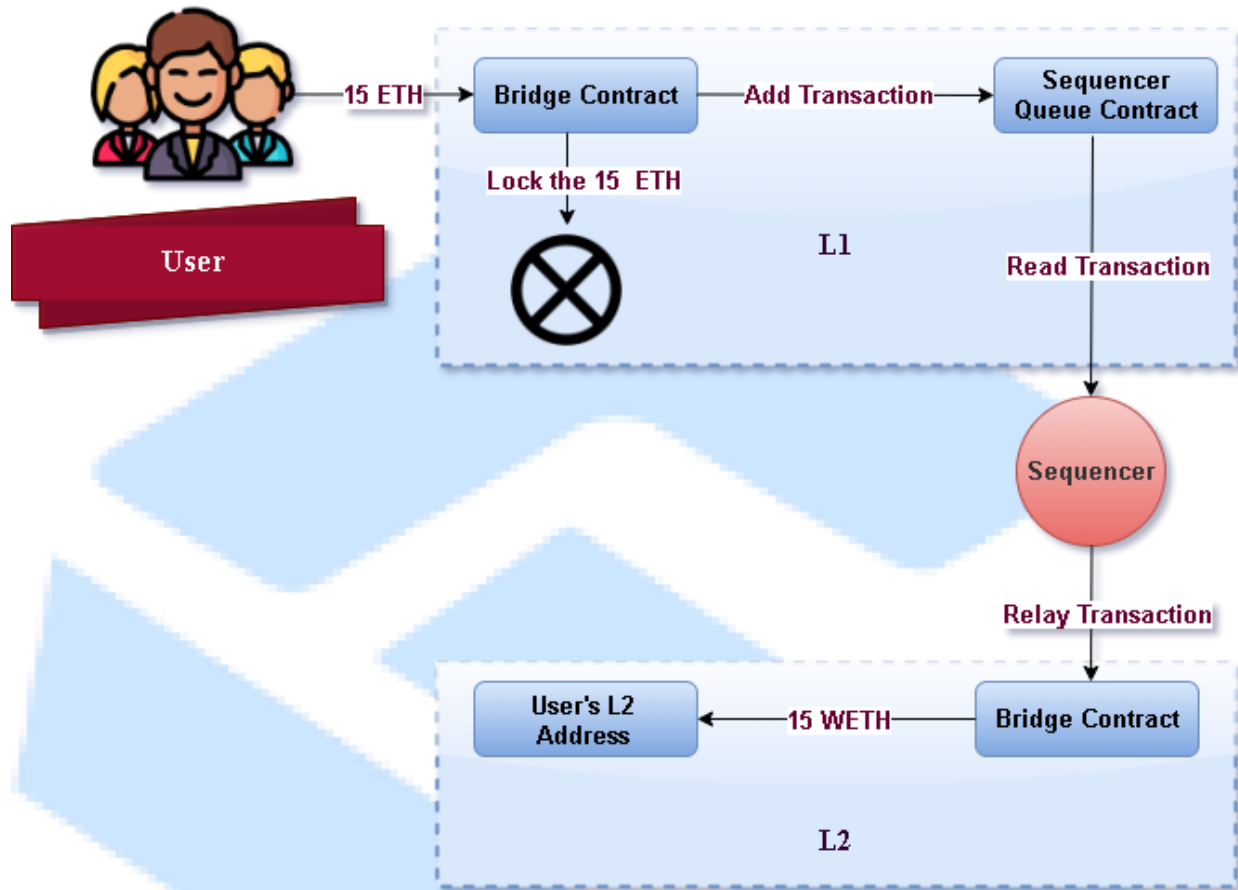
For example, if a user wants to send 10 ETH to their address on L2 so that they can interact with smart contracts on L2, the following steps will happen:

- The user sends 15 ETH to a bridge contract on L1.

- The contract locks the ETH on L1.

- The contract also adds the user's transaction to the queue of transactions that the sequencer must add to L2.

- The sequencer processes this transaction and the ETH is successfully deposited to the user's L2 account.

Note: On L2, WETH, an ERC20 wrapped token, has taken the place of ETH. This will aid L1 transaction replay ability, which will be

described later. The user's L2 account will receive a deposit of 15 WETH as a result of the transaction. Additionally, when a token or WETH is sent to

L2, the bridge contract may actually be two smart contracts cooperating. The Optimism team will offer these bridging contracts upon launch.

L2 -> L1 Transactions

Because the L1 chain frequently needs to confirm the validity of the L2 state root following the transaction (which began on L2), transactions from L2 to L1 can be more challenging. In most cases, the user will transmit the transaction to a particular L2 smart contract. Relayer will then read it and transmit it to L1. A JavaScript service that serves as the relayer has been made available by optimism. It communicates with L2 by means of the sequencer and verifier nodes.
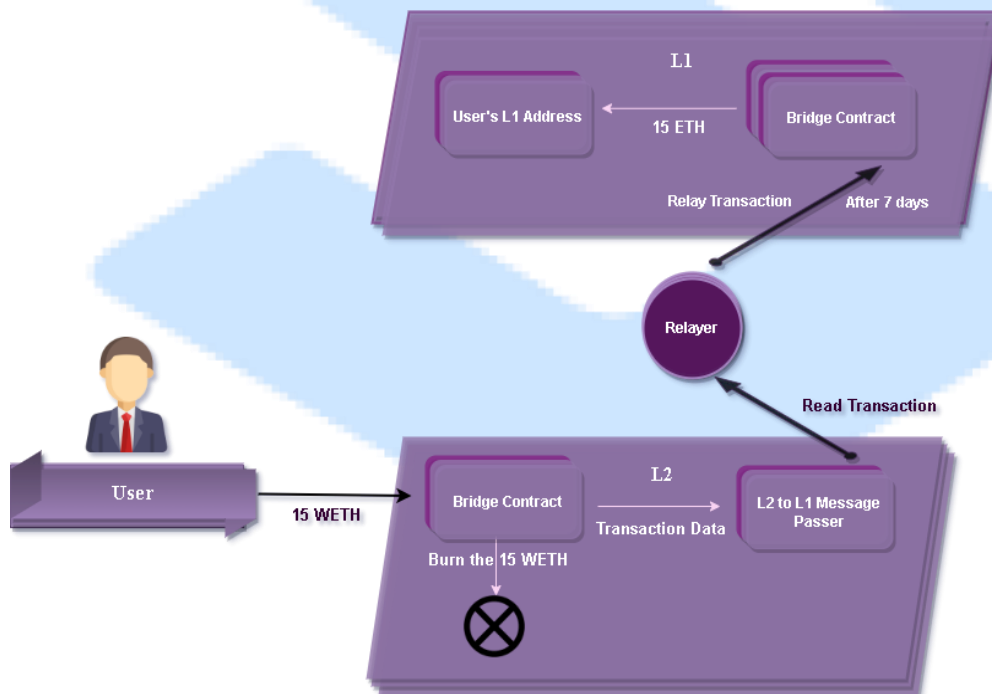
As an illustration of an L2 to L1 transaction, consider what would take place if a user wanted to convert their 15 WETH on L2 address back to ETH on their L1 address.

- 15 WETH are sent by the user to a bridge contract on L2.

- The L2ToL1MessagePasser smart contract receives the transaction details from the bridge contract after burning the WETH. The information for transactions that must be transferred from L2 to L1 is recorded in this smart contract.

- The relayer node gets this transaction data from the L2ToL1MessagePasser and delays forwarding the transaction to L1 for the duration of the fraud proof window (7 days).

- The customer can now withdraw their ETH when the transaction has been processed on L1. The bridge contract that froze their ETH when it was initially transmitted to L2 will be released.

The verifier nodes have adequate time to determine whether the state root reported by the sequencer for this transaction is accurate thanks to this fraud-proof timeframe.

Transaction and state root storage on L1

Since each transaction's transaction data and resultant state root must be saved on L1, it is essential to keep this data's size as little as possible to reduce the system's storage expenses. The steps that each L2 transaction's data is stored on L1 are described below:

- The sequencer creates a batch by combining the calldata for several consecutive L2 transactions.

- The sequencer then delivers this batch to the CanonicalTransactionChain smart contract.

- The smart contract then generates a merkle tree using the hashes of the calldata for each transaction.

- The merkle root of this batch is sent to a smart contract designed for storage by the CanonicalTransactionChain.
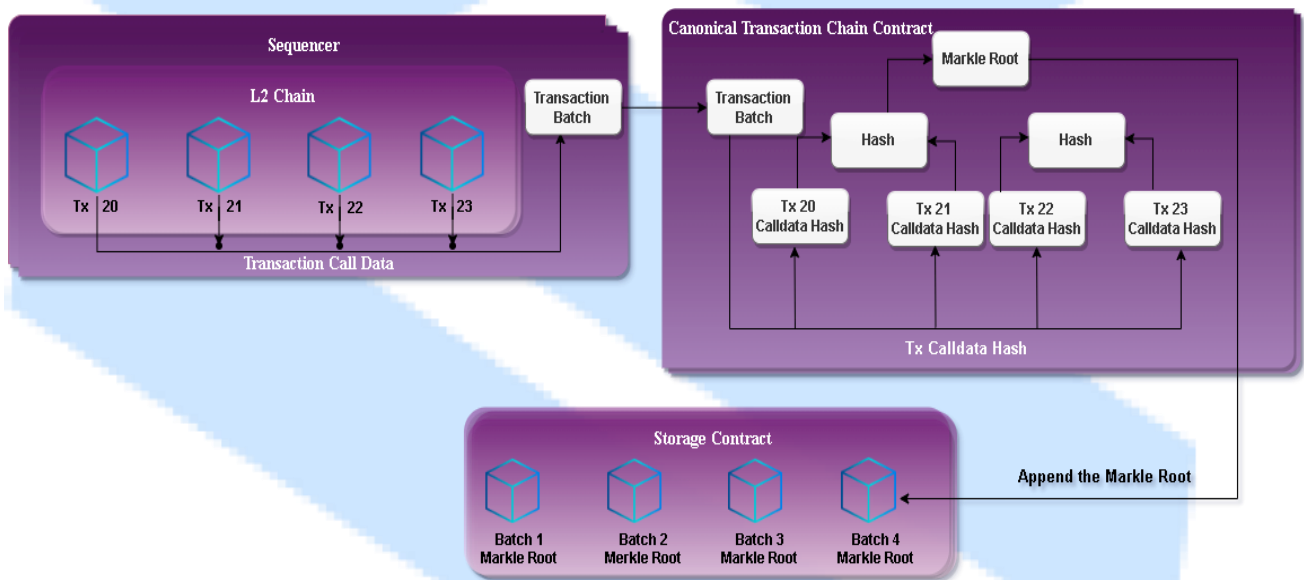
In essence, this procedure involves adding up several transactions, building a Merkle tree, and storing the root. The same procedure also applies to storing state roots. The StateCommitmentChain is the name of the contract that is used to roll up state roots. The insertion of a new merkle root to the storage contract is currently the only modification to the L1 state for a sequence of L2 transactions. As opposed to keeping each transaction independently in the storage contract, this greatly improves the system's efficiency. This procedure is shown in the following diagram.

4.2. Fraud Proofs

Anyone can publish blocks using Sliceledger without having to provide proofs of their validity. Sliceledger provide a time range during which anyone can contest a state transfer, which keeps the chain secure. Sliceledger blocks are therefore referred to as "assertions" because anyone can contest their veracity.

The Sliceledger protocol will start the fraud proof calculation if someone contests an assertion. Every form of fraud evidence is participatory; one person must make a claim before another may refute it. The distinction is in the number of interaction rounds necessary to compute the fraud proof.

To find false statements, single-round interactive proving schemes replay disputed transactions on L1. The computed state root determines who wins the challenge in the Sliceledger protocol, which simulates the re-execution of the disputed transaction on L1 (Ethereum) using a verifier contract. If the challenger is right and the Sliceledger is in the correct state, the operator will be fined and have their bond reduced.

However, posting state commitments for individual transactions and an increase in the amount of data Sliceledger that must be published on-chain are necessary when re-executing transactions on L1 to detect fraud. Repeating transactions consumes a lot of petrol as well. For these reasons, multi-round interactive proving, which more effectively does the same task (that is, identifying invalid Sliceledger operations), is replacing Sliceledger.

4.2.1. Challenge Period

Certainly! In the interaction between Layer 1 (L1) and Layer 2 (L2), messages sent from L2 to L1 cannot be immediately relayed. This waiting period, known as the "challenge period" (lasting one week), serves as a safeguard. During this time, transactions are considered "pending," allowing anyone to challenge their validity.

Optimistic Rollups operate optimistically by posting transaction results to Ethereum without executing them there. Ideally, these results are correct, saving the need for complex and costly operations on Ethereum.

To prevent false results, a "fault proof" mechanism is in place. Published transaction results are subject to the challenge period. Anyone can re-execute the transaction on Ethereum during this time to prove a fault. If successful, the faulty result is discarded, and the challenger may submit the correct result. Financial penalties discourage the dissemination of incorrect results.

Decisions regarding L2 transaction results on L1 should not be made within a smart contract until the challenge period ends. Relaying L2 ⇒ L1 messages using standard messenger contracts is delayed until the full challenge period has passed to avoid acting on potentially invalid results. This cautious approach ensures the reliability and accuracy of transaction outcomes.

4.2.2. Multi-round interactive proving

An L1 verifier contract oversees a back-and-forth procedure between the asserter and challenger during multi-round interactive proving, which eventually determines who is lying. The asserter is required to split the contested assertion into two equal pieces

once an L2 node challenges it. In this situation, each unique claim will have the same number of computing steps as the other.

Next, the challenger will decide which claim it wants to dispute. Up until both parties are debating a claim regarding a single execution step, the dividing procedure (also known as a "bisection protocol") is still in effect. At this point, the L1 contract will settle the conflict by analyzing the instruction (and its outcome) to identify the party that committed fraud.

The asserter must offer a "one-step proof" that demonstrates the accuracy of the contested single-step computation. The challenge is lost if the asserter cannot produce the one-step evidence or if the L1 verifier rejects the proof.

4.2.3. Why fraud proofs matter for Sliceledger

Because they enable trustless finality in Sliceledger, fraud proofs are crucial. A characteristic of Sliceledger called "trustless finality" ensures that a transaction will finally be confirmed, provided it is genuine.

By launching bogus challenges, malicious nodes can attempt to postpone the confirmation of a valid block. However, the legality of the block will finally be established by fraud proofs, leading to its confirmation.

The validity of the chain is dependent on the existence of one honest node, which is related to another security feature of Sliceledger. The truthful node can correctly advance the chain by either posting true claims or refuting false assertions. In any instance, malevolent nodes who disagree with the honest node during the fraud proving procedure will forfeit their stakes.

4.3 L1/L2 Interoperability

Sliceledger enable users to send messages and any type of data between L1 and L2 and are made for interoperability with the Ethereum Mainnet. Additionally, because they are EVM compatible, you can convert current dApps to Sliceledger or construct new dApps using Ethereum development tools.

1. Asset movement

Entering the Sliceledger

Users must deposit ETH, ERC-20 tokens, and other acceptable assets in the Sliceledger bridge contract on L1 in order to use it. The Sliceledger will send a similar amount of assets to the user's specified address after the bridge contract relays the transaction to L2.

User-generated transactions, such as an L1 > L2 deposit, are often queuing up until the sequencer resubmits them to the Sliceledger contract. Sliceledger, on the other hand, permit users to submit a transaction directly to the on-chain Sliceledger contract if it has been delayed for longer than the permitted amount of time in order to maintain censorship resistance.

Some upbeat Sliceledger take a more direct route to avoid consumers being censored by sequencers. Here, in addition to the transactions carried out on the Sliceledger chain, a block is

defined by all transactions submitted to the L1 contract since the previous block (such as deposits). Sequencers cannot postpone user-generated messages once they have been posted on L1 since doing so will cause them to broadcast the (provably) incorrect state root.

Exiting the Sliceledger

Due to the fraud proving system, it is more difficult to withdraw from a Sliceledger to Ethereum. To remove money held in escrow on L1 using an L2 > L1 transaction, a user must wait until the challenge period, which lasts around seven days, is through. However, the withdrawal procedure itself is fairly simple.

The transaction is added to the following batch once the withdrawal request on the L2 rollup is launched, while the user's assets on the rollup are burned. The user can create a Merkle proof to confirm the inclusion of their exit transaction in the block once the batch has been published on Ethereum. Once the delay period has passed, the transaction on L1 can be completed and money can be withdrawn to the Mainnet.

Sliceledger customers can use a liquidity provider (LP) to avoid having to wait a week before withdrawing money to Ethereum. In exchange for a fee, a liquidity provider takes over ownership of a pending L2 withdrawal and pays the user on L1.

Before releasing cash, liquidity providers can verify the legitimacy of the user's withdrawal request (by running the chain

themselves). They are given guarantees that the transaction will finally be verified in this way (i.e., trustless finality).

2. EVM compatibility

The benefit of Sliceledger for developers is their compatibility, or, much better, equivalence, with the Ethereum Virtual Machine (EVM). Sliceledger that support the EVM at the bytecode level are described in the Ethereum Yellow Paper.

EVM-compatibility in Sliceledger has the following benefits:

Without significantly altering codebases, developers can move already-existing smart contracts on Ethereum to Sliceledger chains. This can help development teams implement Ethereum smart contracts on L2 more quickly.

Using Sliceledger, developers and project teams can benefit from Ethereum's infrastructure. This comprises client software, deployment infrastructure, testing tools, programming languages, code libraries, and so on.

Because these tools have been thoroughly inspected, debugged, and refined over time, using current tooling is crucial. Additionally, it eliminates the need for Ethereum engineers to pick up a completely new development stack.

3. Cross-chain contract calls

Users (externally owned accounts) engage with L2 contracts by either submitting a transaction directly to the rollup contract them or delegating the task to a sequencer or Validator. Using bridge contracts to convey messages and pass data between L1 and L2, Sliceledger enable contract accounts on Ethereum to

communicate with L2 contracts. This means that functions from contracts on an L2 Sliceledger can be called from an L1 contract on the Ethereum Mainnet.

Cross-chain contract calls take place asynchronously, which means the call is made first and is then carried out afterwards. On Ethereum, calls between the two contracts create outcomes right away. This is different.

The token deposit mentioned previously is an illustration of a cross-chain contract call. A contract on layer one (L1) escrows the user's tokens and notifies a paired contract on layer two (L2) to mint an equal number of tokens on the rollup.

The sender is often responsible for paying the computation <u>gas charges</u> because cross-chain message calls result in contract execution. To avoid the transaction failing on the target chain, a high petrol limit should be established. A good example is the token bridging scenario; if the L1 side of the transaction—depositing the tokens—works but the L2 side—mining new tokens—fails owing to insufficient gas, the deposit is lost forever.

Last but not least, it is important to remember that L2 > L1 message calls between contracts must take into account delays (L1 > L2 calls are normally completed after a few minutes). This is due to the fact that messages received from the Sliceledger to Mainnet cannot be processed until the challenge window closes.

4.4. Ethereum Scaling

As previously said, Sliceledger assure data availability by publishing compressed transaction data on Ethereum. To scale throughput on Ethereum with Sliceledger, it is essential to have the capacity to compress data broadcast on-chain.

The typical block size on the main Ethereum chain is 15 million gas, and there are gas-based restrictions on the amount of data that may be stored in a block. This limits the amount of gas that each transaction may use, but it also allows us to process more transactions per block by generating less transaction-related data, directly scalability.

Parameter	Ethereum (L1)	Sliceledger (L2)
Nonce	~3	0
Gas price	~8	0-0.5
Gas	3	0-0.5
To	21	4
Value	9	~3
Signature	~68 (2 + 33 + 33)	~0.5
From	0 (recovered from sig)	4
Total	~112 bytes	~12 bytes

These figures can be used to demonstrate the scalability enhancements provided by a Sliceledger using some preliminary calculations:

Every block has a 15 million gas target size, and one byte of data requires 16 gas to verify. The average block may carry 937,500 bytes of data when the average block size is divided by 16 gas (15,000,000/16).

The typical Ethereum block can handle 78,125 Sliceledger transactions (937, 5000/12), or 39 Sliceledger batches (assuming

each batch contains an average of 2,000 transactions), if a basic Sliceledger transaction requires 12 bytes.

The Sliceledger's processing speeds would be around 5,208 transactions per second if Ethereum produces new blocks every 15 seconds. In order to calculate this, divide the maximum number of basic Sliceledger transactions that an Ethereum block can contain (78,125) by the typical block duration (15 seconds).

5. Benefits of Sliceledger

5.1. Scalability

Optimistic Rollups offer significant improvements in Blockchain scalability. By shifting the majority of transaction processing to the Sliceledger chain, these solutions reduce the load on the main chain. This approach enables a substantial increase in transaction throughput. Since the main chain is relieved of the burden of processing every transaction, it can prioritize security, consensus, and the execution of more complex operations. This scalability enhancement is particularly critical for Blockchain like Ethereum, which often face congestion and high gas fees.

5.2. Cost Efficiency

One of the primary benefits of Sliceledger is their potential to reduce transaction fees on the main chain. High fees have been a significant barrier to entry for many users and developers on Blockchain networks. By conducting the majority of transactions on the Sliceledger chain, users can enjoy significantly lower fees. This cost efficiency makes decentralized applications (dApps) more affordable to use and develop, promoting wider adoption and innovation within the Blockchain ecosystem.

5.3. Security

Sliceledger maintain a high level of security through a combination of optimistic execution and fraud proofs. While transactions are initially assumed to be valid without immediate validation, the system relies on validators and users to monitor the Rollup chain for any malicious activity. In case of fraud or invalid transactions, participants can submit Fraud Proofs to challenge the Sliceledger chain's validity.

This dual-layer security approach ensures that only valid transactions are ultimately accepted, preserving the integrity of the Sliceledger chain. Users can have confidence in the security of their transactions and assets, even when conducting most of their activities on the Layer 2 Rollup chain.

5.4. Interoperability

Sliceledger is designed to be versatile and can be applied to various blockchains. This feature enhances cross-chain interoperability, allowing assets and data to move seamlessly between different Blockchain networks. By enabling

interoperability, Sliceledger contribute to a more connected and accessible Blockchain ecosystem.

Interoperability also extends the reach of decentralized applications, making it easier for developers to build cross-chain solutions and for users to access a broader range of services. This interconnectedness is crucial for the growth and maturation of the Blockchain industry, as it breaks down barriers and fosters collaboration between different Blockchain ecosystems

6. Challenges and Limitations

6.1. Data Availability

Data availability is a significant concern for Sliceledger. If Sliceledger's chain data becomes inaccessible or is lost, it can create difficulties during dispute resolution. Ensuring the continuous availability and integrity of Sliceledger data is crucial for the reliability of the system.

6.2. Latency

Sliceledger, while faster than Layer 1, still introduce some latency due to the dispute resolution process. The time required for validating and resolving disputes can impact the responsiveness of the network, potentially affecting user experience in real-time applications.

6.3. User Experience

Sliceledger require users to place trust in the validators of the Sliceledger to maintain network integrity. This reliance on validators may raise trust issues, as the security of the system depends on their honest behavior. Improving user trust and confidence in the Sliceledger validators is essential for widespread adoption and acceptance of this scaling solution.

7. Use Cases

7.1. DeFi Applications

Decentralized Finance (DeFi) has emerged as a transformative force in the world of finance, offering an alternative to traditional

banking systems by providing open and permissionless financial services. However, DeFi platforms often face challenges related to scalability, high gas fees, and network congestion, particularly on Ethereum. Sliceledger present a compelling solution to address these challenges and unlock the full potential of DeFi applications. Here's how DeFi platforms can benefit from Sliceledger:

Scalability:

DeFi platforms are known for their high transaction volume, which can lead to network congestion and slow confirmation times on the main Blockchain. Sliceledger significantly increase the throughput of DeFi applications, enabling them to process a much larger number of transactions per second. This scalability ensures that DeFi platforms can accommodate the growing demand for their services.

Lower Transaction Costs:

High gas fees on Ethereum have been a significant barrier for DeFi users, particularly those with smaller holdings. Sliceledger reduce gas costs, making DeFi accessible to a broader range of users. Lower fees also make it more cost-effective for users to

perform activities like trading, providing liquidity, and participating in lending and borrowing protocols.

Enhanced User Experience:

DeFi users benefit from a smoother and more efficient user experience on Sliceledger based platforms. Transactions are processed quickly, and users do not need to wait for extended

confirmation times, resulting in a more seamless interaction with DeFi services.

Liquidity and Trading:

Sliceledger improve the liquidity and trading experience on decentralized exchanges (DEXs) and automated market makers (AMMs). Users can swap assets, provide liquidity to liquidity pools, and engage in complex trading strategies with reduced costs and minimal slippage.

Lending and Borrowing:

DeFi lending and borrowing platforms can leverage Sliceledger to make borrowing assets more cost-effective. Lower transaction fees and faster transaction confirmations benefit borrowers who seek quick access to funds and lenders looking to maximize their returns.

Cross-Platform Interoperability:

Sliceledger networks can potentially bridge with other compatible Layer-2 solutions or main blockchains. This allows for cross-platform asset transfers and interactions, expanding the reach and interoperability of DeFi applications.

Security:

DeFi platforms on Sliceledger maintain the security and trustlessness of the underlying Blockchain. Users can rely on the Blockchain's security guarantees to protect their assets and ensure that smart contracts operate as intended.

Environmental Sustainability:

Sliceledger, by processing most transactions off-chain, contribute to reducing the environmental impact of DeFi activities, addressing concerns related to Blockchain's energy consumption.

7.2. Gaming

Blockchain-based games and virtual worlds have gained immense popularity, but they often face scalability issues and high transaction costs. Sliceledger offer several advantages for this use case:

Fast and Affordable Transactions:

In-game transactions, such as buying, selling, or trading virtual assets, can be processed quickly and affordably on Sliceledger networks. This enhances the gaming experience by reducing latency and transaction fees.

Scalable Virtual Economies:

Virtual economies within games and virtual worlds can expand without worrying about Blockchain congestion. This scalability encourages the creation of larger and more complex virtual ecosystems.

Cross-Game Asset Interoperability:

Users can potentially transfer assets between different Blockchain-based games and virtual worlds that utilize the same Sliceledger network. This interoperability could lead to unique and innovative gameplay experiences.

Enhanced Security:

The security provided by the underlying Blockchain ensures that in-game assets and virtual land ownership are secure and tamper-proof. Players have greater trust in the scarcity and ownership of digital assets.

Monetization Opportunities:

Game developers can tokenize in-game assets, such as skins, characters, or items, as NFTs on Sliceledger networks. This opens up new revenue streams for developers and players alike.

7.3. Supply Chain and IoT

Supply chain management and IoT (Internet of Things) applications can harness the potential of Blockchain technology while overcoming scalability challenges through Sliceledger:

Transparent and Immutable Records:

Sliceledger offers transparent and immutable records of goods and events in supply chains and IoT networks. This transparency reduces fraud and ensures the integrity of data.

Scalability for IoT Data:

IoT devices generate vast amounts of data. Optimistic Rollups can handle this data efficiently, allowing for real-time tracking, monitoring, and analysis of IoT-generated information.

Decentralized Data Sharing:

Multiple parties involved in supply chains can securely and efficiently share data through Sliceledger-based smart contracts.

This reduces the reliance on centralized intermediaries and enhances trust between stakeholders.

Reduced Costs:

Traditional supply chain and IoT systems involve multiple intermediaries, paperwork, and reconciliation processes. Sliceledger streamline these processes, reducing operational costs.

Compliance and Auditing:

Regulatory compliance and auditing processes can be automated on the Blockchain. This ensures that supply chain and IoT operations adhere to industry standards and legal requirements.

8. Conclusion

Sliceledger offer a compelling solution to Blockchain scalability issues, providing benefits such as scalability, cost-efficiency, security, and interoperability. However, they also come with challenges related to data availability, latency, and user trust. As Blockchain technology continues to evolve, Sliceledger are likely to play a significant role in enabling the mass adoption of decentralized applications across various industries.