Decentralizing Sequencers in Rollups using Delegated Proof-of-Stake Consensus Mechanism

Md Zaki Muzahid
Waseda University
205, TOKYO β Narimasu 12, 6-36-4, Akatsuka, Itabashi-ku, Tokyo, 175-0092 Japan

ABSTRACT
In the Ethereum blockchain network, high transaction fees due to limited block space and high demand necessitate scalable solutions. Layer 2 (L2) scaling solutions, particularly rollups, offer a promising approach by processing transactions off-chain and posting compressed data to the main chain (Layer 1). However, current L2 rollups rely heavily on centralized sequencer nodes, which introduces centralization risks and single points of failure. Thus, to address these concerns, this paper explores the existing issues associated with centralized sequencers exemplified by real-life incidents. Consequently, reviews the existing decentralized sequencer models by describing their operations. In addition, this study proposes a novel approach of decentralizing sequencers leveraging the Delegated Proof of Stake (DPoS) consensus mechanism depicting its' components and step by step procedures. Finally, providing comparison among the novel approach and the existing decentralized sequencer frameworks along with their limitations.

General Terms
Cybersecurity, Decentralization, Blockchain, Consensus Mechanism.

Keywords
Rollups, Decentralized Sequencers, Layer 2 (L2) Scaling, Delegated Proof of Stake.

1. INTRODUCTION
The Ethereum blockchain network faces scalability issues due to its limited block space and high demand, leading to high transaction fees [1]. The transition to Layer 2 solutions, such as rollups, has been pivotal in addressing these issues by increasing transaction throughput and reducing costs. Rollups process transactions off-chain and then submit the compressed transaction data back to the Ethereum mainnet. However, the reliance on a single sequencer node to order and batch transactions poses centralization risks and potential network vulnerabilities.

To mitigate these issues, there has been a growing interest in decentralized sequencer models. This paper introduces a novel approach to decentralized transaction sequencing by leveraging the Delegated Proof of Stake (DPoS) consensus mechanism. The proposed DPoS-Based Decentralized Sequencer model will be compared with existing decentralized sequencing designs, and the trade-offs will be analyzed.

The Delegated Proof of Stake (DPoS) consensus mechanism enhances efficiency and democratic governance in blockchain networks, providing a solid foundation for decentralizing sequencing. In DPoS, participants stake cryptocurrency in a pool to vote for delegates responsible for maintaining the network. Delegates create and sign blocks, validate transactions, and perform essential functions. Its democratic and decentralized structure makes DPoS well-suited for sequencing in rollup solutions.

This paper is structured as follows: Section 1 defines the problem with centralized sequencers, while Section 2 provides background on rollups, sequencers, and the DPoS consensus mechanism. Section 3 reviews existing research on decentralized sequencers, and Section 4 presents the proposed DPoS-Based Decentralized Sequencers model. The design features five sequencers elected via a transparent voting process, rotating across epochs to ensure continuous and decentralized transaction ordering. The head sequencer orders transactions using a First-Come-First-Serve policy, while others verify and commit them, providing secure and efficient processing. Potential drawbacks, including Sybil attacks and vote bribery, are also discussed. Lastly, comparative analysis was conducted with existing research in section 5. Key criteria, including sequencing policy, decentralization, data availability, sequencer management, and incentive were examined, with addition to comparison of limitations.

Problem Definition: Despite the advantages of rollups, the sequencer's critical role in transaction ordering and batching means that any failure or misbehavior can adversely affect the entire network. Real-life incidents with Polygon zkEVM and Arbitrum illustrate these vulnerabilities, where sequencer failures led to network downtime and transaction delays. Having one node to perform the whole sequencing job makes it a more centralized approach as the sequencer node holds a significant amount of importance. This has significant potential to result in Single Point of Failure, meaning if the sequencer fails to perform or appear absent during their job, there will be network downtime, and delay for users' transaction processing.

Thus, to define the problem: *The current approach of utilizing a single sequencer node introduces significant centralization risks. This centralized control not only undermines the decentralized ethos of blockchain technology but also creates a single point of failure. If the sequencer node fails to perform its duties or becomes unavailable, the downtime occurring from this essentially affects the entire network and delays in transaction processing thus affecting users. While efforts to decentralize sequencers have been made, they come with their own style of operations and trade-offs. In essence, it is crucial to develop a solution that negates centralization and single points of failures associated with centralized sequencers.*

Polygon zkEVM network downtime: On April 08, 2024, Polygon zkEVM Mainnet Beta experienced a significant network downtime due to an L1 reorganization (reorg) that led to a state synchronization issue [2]. The incident highlighted vulnerabilities within the network's sequencer and synchronizer mechanisms, necessitating immediate recomputation of the network's state. Specifically, due to the sequencer failing to update the state, it caused the state to differ. Meaning, this resulted in a discrepancy between the actual state

of the blockchain (the trusted state) and the state as perceived by the sequencer's client (the virtual state), resulting in some transactions not being executed or having incorrect execution.

Arbitrum network downtime: On January 9, 2022, Arbitrum, a Layer 2 scaling solution for Ethereum, experienced a 10-hour outage due to a hardware failure in its Sequencer node [3]. The Sequencer, which orders transactions on the network, encountered a problem causing the network to go offline. During the outage, 284 transactions were accepted but logged only after the network rebooted. No funds were lost, and the network has since resumed normal operations. The incident highlighted the current centralization of the Sequencer, which Offchain Labs, the developer of Arbitrum, acknowledged and is working towards decentralizing to minimize future downtimes.

On December 15th, Arbitrum One faced a sequencer outage that disrupted transaction processing and affected gas pricing [4]. The outage was caused by a backlog in the batch poster due to an Ethereum client issue and a high volume of small transactions (inscriptions). This failure affected the sequencer's feed, disconnecting third-party nodes and the public RPC fleet, resulting in delays and failed transactions.

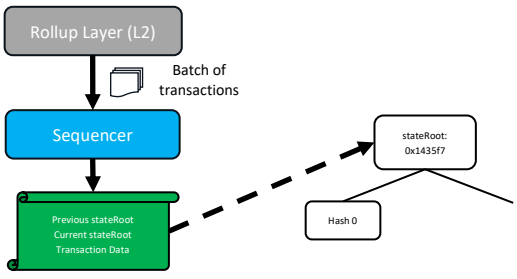
2. BACKGROUND

2.1 Rollups

Rollups are a key Layer 2 (L2) scaling solution for Ethereum, executing transactions off-chain to reduce the computational and storage load on Layer 1 (L1) while maintaining security and decentralization [5]. In this architecture, a Sequencer orders and batches transactions before submitting them to the L1 rollup smart contract, which maintains the current state.

The state consists of accounts—Externally Owned Accounts (EOAs), controlled by private keys, and Contract Accounts, governed by smart contract logic, each with fields for nonce, balance, code, and storage [6]. Rollups manage these states off-chain, storing a "state root" representing the entire state. Transactions update the state root, but for scalability, multiple updates are batched, and only the aggregated state root is committed to the main chain, allowing efficient off-chain processing while maintaining on-chain integrity. This approach reduces the amount of data that needs to be recorded on the main chain, improving scalability.

A batch submitted to the rollup contract includes a compressed representation of the batch of transactions, the previous state root (The previous state root is the cryptographic hash that represents the state of the rollup layer at the end of the most recent finalized batch of transactions before the current batch is applied), and the current state root reflects the post-batch state.



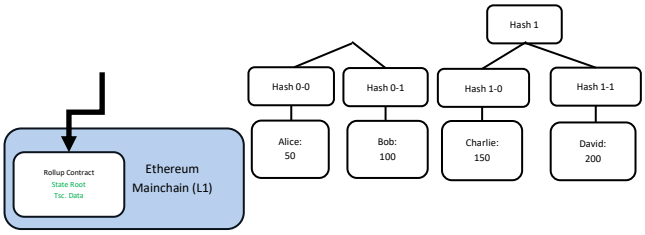


Fig 1: Rollups structure in Ethereum

The L1 rollup contract verifies that the previous state root in a batch matches the current state root, ensuring transaction consistency, and then updates to the new state root. To reduce gas fees, batch transaction data is compressed and sent as a read-only calldata parameter in Solidity. Unlike persistent storage, calldata is transient and does not occupy blockchain storage, making it a far cheaper option than writing data on chain, which is computationally intensive and costly.

2.2 Sequencers: The Centralized Nodes in Rollup Architecture

In rollup architectures, sequencers are centralized nodes that order Layer 2 (L2) transactions and batch them for submission to the Layer 1 (L1) rollup contract, as seen in Polygon zkEVM. They ensure L2 state changes are accurately reflected on L1 but cannot alter transactions, since these are signed by users' private keys. In Optimistic rollups, altered transactions can be detected by verifiers.

However, sequencers can censor transactions (selective filtering) or reorder them to enable front-running attacks. In front-running, a sequencer inserts its own transaction ahead of a target's. For example, in the "FairWin" gambling platform [7], an attacker pre-empted a victim's investment transaction by using the same invite code with a smaller deposit, redirecting future rewards to themselves and stealing the victim's intended funds. If a sequencer becomes unresponsive, transaction ordering and batch submission halt, causing delays, backlogs, and degraded network performance.

An illustrative example of the sequencer process can be seen in the Polygon zkEVM rollup solution. As seen on Fig. 2, the Sequencer employs a structured approach to manage transaction batches using the BatchData structure (a Struct type - user-defined data types that allow to group multiple variables of different types under a single name, basically making it easier to manage and organize data in the smart contracts) in Solidity [8]. "Batchdata" differs from "calldata" as it is used to organize and manage the transactions and their associated metadata. When the rollup contract receives a batch submission, the batch data is passed as function arguments via "calldata". This means the batch information is temporarily stored in the "calldata" during execution of the function that processes the batch.

Fig 2: Polygon ZkEVM Rollup Structure

This process involves two key steps illustrated in Fig. 3:

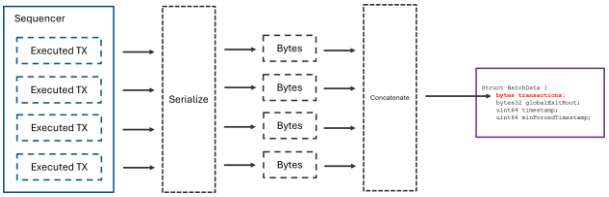


Fig 3: Polygon ZkEVM Sequencing process

1. **Serialization**: Initially, the Sequencer serializes the transactions using Recursive Length Prefix (RLP) encoding. Serialization converts various data types into a byte format, which is essential for the subsequent concatenation step.
2. **Concatenation**: Following serialization, the Sequencer concatenates the serialized transactions into a single, long byte string. This byte string is then stored in the ”transactions” variable bytes field of the BatchData structure.

Once the transactions are serialized and concatenated, the Sequencer calls the sequenceBatches function in the Polygon rollup smart contract on L1. This function reads the concatenated byte string from the BatchData structure, thereby transferring the batched transactions (utilizing ”calldata” in the function) to the L1 rollup contract [9].

This structured approach not only enhances the efficiency of transaction batching but also exemplifies the meticulous processes undertaken by Sequencers to ensure the accurate and secure transfer of transaction data from L2 to L1.

2.3 Delegated Proof-Of-Stake

The Delegated Proof of Stake (DPoS), proposed by Daniel Larimer [10], is a more efficient variant of Proof of Stake, enabling stakeholders to vote for delegates who validate blocks. Typically, 21–101 delegates are elected, with voting power proportional to stake; smaller stakeholders may delegate their votes. Delegates take turns producing blocks in assigned time slots, and misconduct or poor performance results in loss of reputation and stake. In addition to block production, delegates govern the chain, verify fees, and maintain network integrity [11].

As shown in Fig. 4, stakeholders may pool stakes to increase voting influence, either voting directly or delegating votes. Candidates submit proposals outlining qualifications and plans, and those with the most votes become delegates. Elected delegates create and sign blocks, validate transactions, verify peers' work, and share a portion of their rewards which is typically transaction fees—with supporting voters.

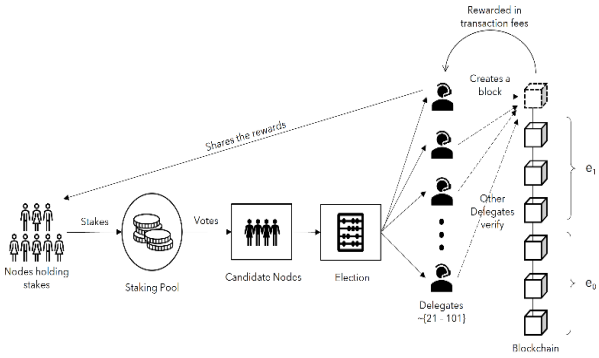


Fig 4: Delegated Proof-Of-Stake Overview

3. EXISTING RESEARCH

3.1 Espresso Sequencer

The Espresso Sequencer utilizes the protocol named ”HotShot,” a Byzantine Fault Tolerant (BFT) consensus protocol designed to decentralize participation within the sequencer network, offering high throughput and rapid finality [12][13].

3.2 Astria: The Shared Sequencer

Astria represents a shared sequencing network designed to replace centralized sequencers with a decentralized alternative; by allowing multiple rollup contracts to share a single network of sequencers, Astria aims to provide censorship resistance, fast block confirmations (finality), and atomic cross-rollup composability [14][15].

3.3 Metis Decentralized Sequencer

Metis introduces a Decentralized Sequencer Pool (DSP) to achieve decentralization in Layer 2 networks, mitigating the risks of single-point failures associated with centralized sequencers [16]. The DSP employs a Proof of Stake (PoS) like mechanism for sequencer selection, wherein sequencers with high staked tokens are gathered into a pool. Users initiate transactions, which are then sent to the sequencer for validation and batching into block before being submitted to the rollup contract in Layer 1 [17]. The consensus mechanism employed by Metis is Tendermint, a Byzantine Fault Tolerance (BFT) protocol that allows the network to function even if up to one-third of nodes fail, including those that act maliciously.

3.4 Fernet Sequencer

Fernet sequencer protocol developed by Aztec Labs - a company specializing in privacy-focused technologies for blockchain applications known for developing layer 2 solutions such as rollups and smart contracts on public blockchains, particularly Ethereum, is designed to ensure random sequencer selection, enhancing decentralization and fairness in transaction processing. In each iteration, a Verifiable Random Function (VRF) assigns a secret score to each sequencer, used to rank them [18].

3.5 Radius Sequencer

Radius Sequencer introduces a Shared Sequencing Layer utilizing an encrypted mempool and Practical Verifiable Delay Encryption alongside zkbased schemes [19]. It separates transaction roles into ordering, executing, and proofing phases. Users send encrypted transactions with a time-lock puzzle and zk-proof (SNARK) to the sequencer, which validates and orders them, providing pre-confirmation before decryption and transmission to Layer 1 (L1) [20].

4. PROPOSED SOLUTION

Before introducing the novel approach to decentralizing sequencer model, first the motivation behind the design and the requirements needs to be stated.

4.1 Motivation

The motivation is: *To decentralize the sequencing process in rollup solutions to mitigate the risks associated with single points of failure and centralization; aligning with the decentralized ethos of blockchain technology, and to ensure that users do not experience any delay in their processing of transactions due to network downtime.*

4.2 Requirements

The requirements that need to be met for a decentralized sequencer design are as follows. The criteria are: Sequencing

Policy, Decentralization, Data Availability, Sequencer Management, and Incentive.

Criteria:

1. **Sequencing policy:** The Sequencing Policy should be fair sequencing. It is deemed "fair" if there is no selection of transactions according to some bias such as having priority to select some transactions over others, and the ordering scheme utilized in the policy should reflect this while providing resistance to potential censorship.
2. **Decentralization:** Decentralization aims to distribute the sequencing process across multiple nodes rather than relying on a single entity. This requires multiple sequencers forming a network, where a consensus mechanism allows the nodes to collectively agree on the ordering of transactions. By achieving agreement through consensus, the sequencing process becomes more robust, transparent, and resistant to centralization.
3. **Data Availability:** Ensuring data availability is a critical requirement for the design of a decentralized sequencer model, as it guarantees that all nodes within the sequencer network have access to the complete set of the transactions. Data availability facilitates the consistent ordering and validation of transactions across different sequences by ensuring that all the nodes are able to receive the same transaction data.
4. **Sequencer Management:** This requirement defines the protocol for selecting a new head sequencer if the current head is absent during an epoch. It also specifies the consequences for sequencers that act maliciously or fail to perform their duties, including a slashing mechanism to penalize misbehavior. Such measures discourage dishonest actions and help maintain the integrity and reliability of the network.
5. **Incentive:** This requirement to provide incentive for sequencers to perform their duties for the contribution to the protocol, as without incentive, there will be lack of engagement and participation.

4.3 Delegated-Proof-of-Stake based Decentralized Sequencers

The novel design towards decentralizing sequencers introduced in this paper is Delegated-Proof-of-Stake Decentralized Sequencers. This design utilizes the DPoS consensus mechanism for decentralization. In Delegated Proof of Stake (DPoS) consensus mechanisms, nodes in the network stake their currency in a pool to vote for as delegates. Elected delegates are tasked with network maintenance, including block creation, transaction validation, and related duties. Now, using the same idea, the sequencers will be decentralized using DPoS. By looking at the model of DPoS consensus mechanism, it is a very suitable model for decentralizing sequencers because in DPoS there is a set

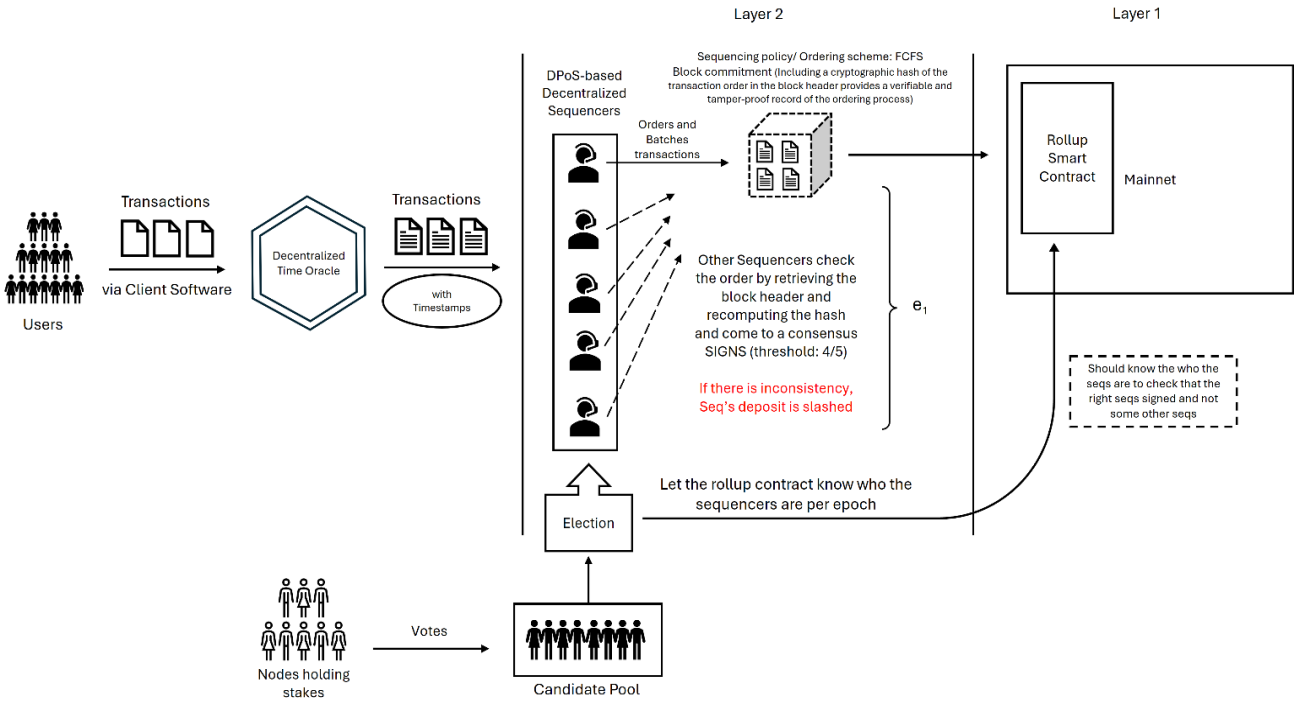


Fig 5: DPoS Based Decentralized Sequencers Overview

number of delegates who creates the block and others verify it; this idea seems ideal for the design in decentralizing sequencers as a set amount of sequencer can order the transactions and verified by other sequencers per epoch.

The flow of the design will follow this procedure illustrated in Figure 5, where users will initiate transactions via their digital wallets (such as MetaMask) using client software (such as Geth). To enforce First-come-first-serve ordering, there is a requirement to timestamp these transactions.

To accomplish this, a Decentralized Time Oracle comprising multiple trusted independent nodes that provide tamper-proof timestamps—is used, such as the Witnet Oracle, which offers a "timestamp" function to record transactions in UTC format [21]. When a transaction is received, the oracle generates a timestamp that is attached to the transaction. The transaction is then sent to the mempool, a distributed waiting area maintained by each node in the network, including sequencers. Sequencers fetch transactions from their local mempool [22] and order them chronologically based on the assigned timestamps,

ensuring a consistent and verifiable transaction sequence across the network.

A group of five sequencers is elected each epoch, with one sequencer rotating out in the next epoch. At the end of voting, the top five candidates with the most votes are recorded in the L1 rollup smart contract along with their deposits. This model follows a "one-voter-one-vote" principle to ensure equal voting power. The head sequencer orders transactions based on submission timestamps using a First-Come-First-Serve policy, committing a cryptographic hash of the transaction order to the block header. The remaining sequencers verify the order by recalculating the hash and reaching consensus through a peer-to-peer protocol (e.g., libp2p). If fewer than four out of five sequencers agree, the block is unsigned, the head sequencer's deposit is slashed, and the batch is reprocessed by the next sequencers.

Network Time Protocol (NTP) synchronizes sequencers, and verified blocks are forwarded to the L1 rollup contract. Transaction data is stored in the cost-efficient, read-only calldata parameter, providing accessible data availability for all network nodes.

Furthermore, the structure of DPoS' election process to elect few delegates to mine blocks seems to be a very suitable design archetype in the case of decentralizing sequencers, as sequencers are essentially delegating the scalability in transaction batching. So, prospective sequencers must first declare their candidacy before becoming sequencers. In DPoS, this process happens publicly on blockchain forums, often involving proposals detailing qualifications, contribution plans, and why they deserve to be delegates. This "candidacy" proposal will follow the same process in this design where users will be required to express their suitability for becoming a sequencer. The candidates will be joining a candidate pool, accessible for user voting, where these candidates will become sequencer according to the votes they receive.

For the voting mechanism, a Layer 2 voting smart contract is proposed. Stakeholders in the DPoS system cast votes by calling the contract's vote function and specifying their chosen candidate. Each voter is allocated a single vote, and the contract prevents repeated voting by checking if the voter has already participated. Upon a valid vote, the candidate's vote count is incremented, the voter's address is recorded, and the voting event is logged for transparency. To ensure broad participation, an API will be provided, enabling client applications to securely interact with the contract and allow stakeholders to vote via their digital wallets. This design ensures a decentralized, transparent, and accessible voting process.

Additionally, all sequencers must deposit a certain amount of funds in the L1 rollup smart contract, which is modified to handle deposits, implement a slashing mechanism, and distribute rewards. These deposits serve both as a commitment to the network and as compensation in case of downtime or malicious behavior, incentivizing honest participation. If a sequencer fails to perform their duties during an epoch or is found acting maliciously, their deposit is slashed. After completing sequencing in each epoch, a cooldown period of seven days is enforced, preventing the sequencer from immediately running in the next election and allowing other candidates a fair opportunity to participate.

In essence, this design satisfies all the key requirements. The sequencing policy is "fair," following a First Come First Serve approach that avoids bias in transaction processing. The system is decentralized, distributing sequencing across multiple nodes and verifying their actions via the DPoS consensus mechanism. Sequencer selection is inclusive, as any node can participate in elections, and all voters have equal voting power. Incentives further encourage participation: sequencers earn a portion of transaction fees for their work, and voters supporting elected sequencers also receive rewards at the end of each epoch. This mechanism promotes active engagement and ensures both fairness and efficiency in the network.

4.3.1 Drawbacks

One might argue that adding a consensus mechanism introduces extra complexity, overhead, and potential delays in Layer 2, which could challenge the fast finality and lightweight nature of rollups. Implementing Delegated Proof of Stake (DPoS) in L2 may appear counterproductive given the high costs of L1 operations. However, while DPoS adds some complexity, it improves scalability, security, and economic efficiency.

Security benefits arise from reduced centralization risk in the sequencer role, lowering the chance of malicious behavior and single points of failure. Scalability improves as DPoS enables higher throughput, and economic efficiency is achieved by limiting consensus to a selected group of sequencers, reducing network-wide computational demands. Importantly, despite the added complexity, transaction throughput - the core property of rollups remains high. Nonetheless, DPoS carries additional drawbacks, including susceptibility to certain attacks:

1. **Potential Sybil Attack**: A Sybil attack occurs when an adversary creates multiple identities to gain excessive influence in the network [23]. In a DPoS election, a user could generate many accounts and potentially have all elected sequencers controlled by a single individual. While this attack is costly and requires luck, it remains possible.
2. **Vote bribery**: This is when the candidates offer monetary incentives to voters to solicit votes, which results in an unfair election. This issue can be difficult to detect, especially if done covertly through off-chain agreements.

A potential solution to the Sybil attack is to resize the number of sequencers elected per epoch. In current design, the sequencer queue space is allocated to maximum of five sequencers, but increasing the number of sequencers will decrease the possibility of the Sybil attack. Thus, the size of the queue is inversely proportional to the possibility of these attacks. As increasing the number of sequencers will make it more expensive to create multiple accounts to perform Sybil attacks but the trade off as that increasing the sequencer queue size will increase the delay finality as it will increase the time to reach the consensus among more sequencer nodes.

5. COMPARISON

This section of the paper will explore the comparison between the existing decentralized sequencer models and DPoS-based Decentralized Sequencers model. The criteria that will be compared are sequencing policy, decentralization, data availability, sequencer management and incentive illustrated on Table 1.

5.1 Sequencing Policy

S The DPoS-based Decentralized Sequencers use a First Come First Serve (FCFS) policy to ensure fair transaction ordering without bias toward high gas fees or other priorities. Unlike other designs that favor lucrative transactions, FCFS prioritizes chronological order, promoting fairness. It also helps prevent front-running, as any attempt to insert a transaction out of order would be detected during consensus via mismatched cryptographic hashes.

5.2 Decentralization

Regarding the decentralization aspect of the design, Espresso uses HotShot BFT and a combinatorial lottery to distribute sequencing among nodes. Astria employs CometBFT for a shared node-based consensus. Metis relies on Tendermint BFT, tolerating up to one-third faulty nodes. Fernet uses VRF-based rotation to give many nodes sequencing opportunities. Radius applies RAFT with an elected leader managing sequencing and followers handling data routing [24]. The DPoS model elects multiple sequencers to agree on transaction ordering, maintaining distributed control.

5.3 Data Availability

For data availability, Espresso uses the Tiramisu layer with Verifiable Information Dispersal (VID), encoding block data into chunks so each node stores only one and can verify availability without full data. Astria and Radius use Celestia's Data Availability Sampling (DAS), enabling light clients to verify availability without downloading full blocks. Metis relies on Ethereum, sending sorted transactions to the "CanonicalTransactionChain" contract for public access. Fernet uses its own DA layer, uploading full block content, though specifications are not disclosed. The DPoS model employs the Layer 1 rollup contract's calldata for low-cost storage with Layer 1 security, while alternative DA layers may reduce costs but weaken security.

Table 1. Comparison between existing decentralized sequencers

Decentralized Sequencers	Sequencing Policy	Decentralization	Data Availability	Sequencer Management	Incentive
Espresso	Priority: Tx with high gas Ordering scheme: Undefined	Hotshot – BFT	Tiramisu	Lottery	Undefined
Astria	Priority: Tx with high gas Ordering scheme: Undefined	CometBFT+PoS	Celestia	Based on staked amount	Reward mechanism with native network tokens
Metis	Priority: Tx with high gas Ordering scheme: Undefined	Tendermint – BFT	Ethereum	PoS – Based on staked amount	Reward mechanism with native network tokens (Metis tokens)
Fernet	Priority: Tx with high gas Ordering scheme: Based on sequencer	No consensus mechanism	Fernet's Dedicated DA Layer	Random Scoring using VRF	Reward Mechanism with native network tokens
Radius	Priority: Tx with high gas Ordering scheme: Undefined	RAFT Algorithm	Celestia	Election	Undefined
DPoS-based Decentralized Sequencers	Priority: Chronological Ordering scheme: First-Come-First-Serve (FCFS)	Delegated Proof-of-Stake	calldata	Election	Reward mechanism with gas fees and portions shared with voters

5.4 Sequencer Management

Transitioning to sequencer management, Espresso uses a combinatorial lottery to randomly select sequencers, with HotShot BFT quickly appointing a new head if the current one is absent; misbehaving sequencers are removed from the pool. Astria and Metis follow PoS rules, selecting the highest stakers and replacing malicious sequencers with the next highest, penalizing them by slashing staked funds. Fernet employs a VRF to assign secret scores each round, rotating to the next highest if a sequencer fails; each must stake 16 ETH on Layer 1, which can be slashed. Radius uses elections under the RAFT algorithm, re-electing a leader if absent. The DPoS model also uses elections, shifting leadership if a sequencer is absent and slashing deposits; its "one person, one vote" system ensures equal voting rights regardless of stake.

5.5 Incentive

Lastly, regarding incentive criteria among the designs, Espresso and Radius does not specify what their incentive and reward mechanisms are, while Astria, Metis, and Fernet sequencers earn their respective native token as rewards for their work although the specification about these mechanisms are not elaborated in their respective documentations. The reward mechanism for DPoS-based decentralized sequencers not only compensates sequencers for their work but also allocates a portion of fees to users who voted for the sequencer. This will increase engagement and participation from nodes in the network.

5.6 Comparison of Limitations

After reviewing the characteristics of each decentralized sequencer design, their limitations are summarized in Table 2.

Table 2. Comparison of drawbacks between the decentralized sequencers designs

Decentralized Sequencers	Limitations
Espresso	• Performance Impact of fixed node set • Bottleneck in DA Layer (Mascarpone • Latency Variation with Committee Size • Lack of incentives
Astria	• Complexity (Many interconnected components) • PoS's "Rich gets Richer" problem
Metis	• Reward Inconsistency • Rich gets Richer
Fernet	• Indeterminate Block Proposals • Bribing the Prover Network • Coordination Overhead
Radius	• Unspecified mention of election method • Lack of incentives
DPoS-based Decentralized Sequencers	• Potential Sybil Attack • Vote bribery

Espresso exhibits performance drawbacks with a fixed node set—throughput on Hotshot and Tiramisu testnets was lower with 10 nodes than with 100+, due to limited task distribution. Mascarpone's DA committee size created a bottleneck, and latency varied with committee size. No incentive mechanism was documented.

Astria's multi-component architecture (shared sequencers, rollups, Composer, Relayer, Conductor, DA layer) adds complexity, risking synchronization and efficiency issues. Its PoS-based network also faces the "rich get richer" problem, allowing wealthier validators to dominate.

Metis [25] showed reward inconsistencies, with participants losing or receiving excess rewards. Its PoS-based sequencer list replaces malicious nodes but is also prone to the "rich get richer" effect.

Fernet [26] has three limitations: indeterminate block proposals, where malicious sequencers may withhold data; prover bribery, where lower-ranked proposals may be favored; and high overhead coordination due to VRF computations, communication, and complex incentives, raising operational costs.

Radius uses elections to select sequencers but lacks transparency on the election process and incentives, risking centralization. Meanwhile, the DPoS-based Decentralized Sequencers face potential Sybil attacks and voter bribery.

Given these limitations, the decision to integrate a specific decentralized sequencer into a rollup solution should be based on a comprehensive analysis of how these factors interact with the system's requirements. One should consider the trade-offs, decentralization, and the overall performance of the system under different operational testcases.

6. CONCLUSION

This study addresses centralization risks and single points of failure in Layer 2 (L2) rollup scaling solutions caused by reliance on centralized sequencer nodes. Five existing decentralized sequencer designs and their approaches to mitigating these issues are examined, emphasizing the need for a more distributed and resilient transaction sequencing mechanism.

A Delegated Proof of Stake (DPoS)-based decentralized sequencer model is proposed, that democratizes transaction ordering through a First-Come-First-Serve policy and incentivizes honest participation via rewards for both sequencers and voters. The democratic election process and sequencer rotation across epochs offer a robust framework to overcome limitations of centralized models.

Selecting a decentralized sequencer requires weighing each model's strengths, weaknesses, and alignment with application goals. The analysis aims to guide developers and stakeholders toward solutions that reduce centralization risks. Future work should explore practical implementation and further refinement to ensure viability in real-world deployments. Future scope of this research includes implementing the proposed DPoS-based decentralized sequencer in real-world blockchain environments to evaluate its performance in terms of scalability, latency, cost, and security. This includes comparative analysis against existing centralized and decentralized models and also testing their ability to handle high transaction volumes. Ultimately, assessing the effectiveness of its democratic election process and voter incentives. Consequently, studies will expand to the integration of the model with existing rollup protocols and investigate the potential trade-offs between decentralization and throughput while observing the long-term sustainability and interoperability across various blockchain systems.

7. REFERENCES

[1] T. A. Alghamdi, R. Khalid and N. Javaid, "A Survey of Blockchain Based Systems: Scalability Issues and Solutions, Applications and Future Challenges," in IEEE Access, vol. 12, pp. 79626-79651, 2024, doi: 10.1109/ACCESS.2024.3408868

[2] Polygon ZKEVM: Network Outage Report(04/08). (2024, April 10). Polygon Community Forum. https://forum.polygon.technology/t/polygon-zkevm-network-outage-report-04-08/13751

[3] Fernau, O. (2022, January 10). Arbitrum goes down citing sequencer problems. *The Defiant*. https://thedefiant.io/news/defi/arbitrum-outage-2

[4] ArbitrumFoundation. (n.d.). docs/postmortems/15 Dec 2023.md at· ArbitrumFoundation/docs. GitHub. https://github.com/ArbitrumFoundation/docs/blob/50ee88b406e6e5f3866b32d147d05a6adb0ab50e/postmortems/15_Dec_2023.md

[5] Buterin, V. (2021, January 5). An incomplete guide to rollups. https://vitalik.eth.limo/general/2021/01/05/rollup.html

[6] Buterin, V. "Ethereum white paper." GitHub repository 1 (2013): 22-23. https://ethereum.org/en/whitepaper/

[7] Castonguay, P. (2021, December 12). The collapse of FairWin's $125m Ponzi scheme - Philippe Castonguay - medium. Medium. https://medium.com/@PhABC/the-collapse-of-fairwins-125mponzi-scheme-61a66b273420

[8] Polygon Labs. (n.d.). Transaction batching - Polygon Knowledge Layer. https://docs.polygon.technology/zkEVM/architecture/protocol/ transaction-life-cycle/transaction-batching/?h=batchdata

[9] Polygon Labs. (n.d.-a). Sequencing batches - Polygon Knowledge Layer. https://docs.polygon.technology/zkEVM/architecture/protocol/sequencing-batches/?h=sequen\#sending-batches-to-l1\

[10] Larimer, Daniel. "Delegated proof-of-stake (dpos)." Bitshare whitepaper 81 (2014): 85. http://107.170.30.182/security/delegated-proof-ofstake.php

[11] Yang, Fan et al. "Delegated Proof of Stake With Downgrade: A Secure and Efficient Blockchain Consensus Algorithm With Downgrade Mechanism". IEEE Access (2019). vol 7, pages 118541-118555

[12] EspressoSystems. (2023, September 7.). HotShot/docs/espressosequencer-paper.pdf at main · EspressoSystems/HotShot. GitHub. https://github.com/EspressoSystems/HotShot/blob/main/docs/espresso-sequencer-paper.pdf

[13] Espresso. (2023, September 7). Sequencer Marketplace — Espresso. https://docs.espressosys.com/sequencer/espressoarchitecture/sequencer-marketplace

[14] Astria. (2023, December 11). Introduction — Astria. Astria the Sequencing Layer. https://docs.astria.org/overview/1-introduction

[15] Astria. (2023, December 10). Transaction Flow — Astria. https://docs.astria.org/overview/transaction-flow

[16] Metis Foundation. (2024, March 14). Decentralized Sequencer - MetisSmart L2. https://www.metis.io/decentralized-sequencer

[17] Metis Foundation. (2024b, March 14). Transaction cycle — Metis. Metis Developer Documentation. https://docs.metis.io/dev/decentralizedsequencer/overview/transaction-cycle

[18] Aztec. (2023, October 10). Fernet - A protocol for random sequencer selection for the Aztec Network. HackMD . https://hackmd.io/@aztec-network/fernet

[19] Radius Sequencer. (2023, November 30). Radius - Overview. TheRadius. https://docs.theradius.xyz/deep-dive/introduction

[20] Radius. (2023a, November 30). Encrypted Mempool — Radius. Radius Sequencing. https://docs.theradius.xyz/testnets/porticotestnet/encrypted-mempool

[21] Witnet Decentralized Oracle. (2017, November). Wallet API Endpoints. Witnet Docs. https://docs.witnet.io/developerreference/integrations/wallet-api#send transaction

[22] GeeksforGeeks. (2023, March 28). What is Ethereum Mempool? GeeksforGeeks. https://www.geeksforgeeks.org/what-is-ethereum-mempool/

[23] Douceur, J.R. "The Sybil Attack". Druschel, P., Kaashoek, F., Rowstron, A. (eds) Peer-to-Peer Systems. IPTPS 2002. Lecture Notes in Computer Science, vol 2429. Springer, Berlin, Heidelberg.

[24] Radius. (2023b, November 30). Leader-based — Radius. Radius Sequencing. https://docs.theradius.xyz/testnets/porticotestnet/distributed-sequencing/leader-based

[25] Metis Lab Foundation Security Audit Report. (n.d.). Google Docs. https://drive.google.com/file/d/1AHDVzVUcRh8ghmfLR8qRfaHpgML7v9vW/view

[26] Fernet. (2023, October). Sequencer selection Fernet - HackMD. HackMD. https://hackmd.io/0FwyoEjKSUiHQsmowXnJPw#Introduction