Early Observations of Based Rollups: A Case Study of Taiko

Jan Gorzny, Phillip Kemper, and Martin Derka

Zircuit
{jan, phillip, martin}@zircuit.com

Abstract. Rollups have become popular layer two networks for general purpose layer one blockchains that support smart contracts. In this work, we explore so-called *based rollups*: rollups which use the layer one network for sequencing. First, we explicate the benefits and potential pitfalls of this approach over centralized sequencers. Second, we use the Taiko rollup as a case study of early based rollups. In this case study, we analyze the cost and performance of Taiko and compare the findings to a similar-but-centralized rollup. Finally, we evaluate whether or not the purported properties of based sequencing are apparent in early systems based on the data collected in our case study. We find that while based rollups have promising properties, additional work is needed to make them competitive with centralized sequencers.

1 Introduction

Blockchains like Ethereum [1] have evolved to be dependable and stable and to support applications built on top of them, but not necessarily to maximize transaction throughput. To overcome this limitation, so-called *layer two* (L2) solutions [2] are built as applications on Ethereum, which is called a *layer one* (L1) in turn. Although several L2 designs exist, rollups have emerged as popular solution. As of 6 January 2025, there are 135 active rollups (or variants) which collectively hold more than $56 billion USD in cryptocurrencies [3].

Rollups aim to increase the transaction throughput of the L1 ecosystem. They do this by separating execution from consensus and performing state updates off-chain, posting verifiable data and updates which are considered final after some conditions are met. The execution allows transactions to be ordered and blocks to be built for the L2 – namely, sequencing – in a way that allows blocks to be larger, produced more often, or both. This work studies rollups which use their L1 network to drive sequencing. These rollups are called *based rollups* by Drake, who defines them as follows [4]:

A rollup is said to be based, or L1-sequenced, when its sequencing is driven by the base L1. More concretely, a based rollup is one where the next L1 proposer may, in collaboration with L1 searchers and builders, permissionlessly include the next rollup block as part of the next L1 block.

Based rollups may benefit from improved cross-chain communication and message passing, inherit strong liveness guarantees from the underlying layer one, and are purported to have simpler designs. In this work, we investigate these claims and explicate the design of such a rollup by analysing the Taiko based zero-knowledge (ZK) rollup [5]. Using data from Ethereum and Taiko, we evaluate the cost and performance impact of implementing a based rollup over other designs.

Contributions. This paper makes the following contributions:

- We explicate the advantages and disadvantages of based sequencer rollup designs (Section 2).
- We report on the cost and performance of based sequencing by analyzing data obtained from the Taiko based rollup. We compare this to a non-based ZK rollup, Scroll [6], to illustrate the differences in performance that may arise as a result of based sequencing (Section 3).
- We evaluate the advantages of based rollups in the context of the data collected (Section 4). We provide the code and data collected for future research.

1.1 Related Work

Rollups (a.k.a *commit-chains* [7] or *validating bridges* [8]) have received a lot of research attention. A general introduction to rollups can be found in works like [9], [2], and [10]. In [10], various topics like based sequencing are used to build a taxonomy that can compare and differentiate modern rollups. Others (Motepalli et al. [11], Mamageishvili and Schlegel [12]) have explored decentralized sequencers and shared sequencers. Both of these concepts change how a rollup's sequencer behaves, but a decentralized sequencer may not be a based one. Incentives for honest validators for optimistic rollups are studied in [13].

The costs and performance of rollups have also been studied. The authors of [14] and [15] have looked at the performance and cost savings of rollups in general and for zero-knowledge rollups like Taiko. The authors of [16] looked at how compression is used by rollups to reduce the cost of posting data on-chain, while the authors of [17] investigated strategies for efficient batch posting itself, and strategies for posting data as blobs (after EIP-4844 [18]) were studied in [19].

To the best of the authors' knowledge, no other works explore and evaluate the performance of based sequencers. We address this gap in order to confirm that the purported advantages of based sequencing are plausible and to measure the trade-offs that are necessary for these designs. Moreover, our work provides a dataset that others can use to analyze early implementations of a based sequencer and we establish research directions for these systems that should be resolved in order to support mature based rollups.

2 Based Sequencing

We describe based sequencing in this section. We focus on the specifics of how a rollup's sequencing is driven by the L1; there may be several ways that this can be

+/−	Trait	Short Justification
+	Liveness	L2 continues to function as long as the underlying L1 continues to function.
+	Decentralization	Permissionless inclusion of the next L2 block.
+	Simplicity	No need for escape hatches (see, e.g. [24]), additional consensus, or signature verification.
+	L1 economic alignment	MEV [25] extraction on based rollups is performed by actors building L2 blocks – namely, L1 actors – and therefore any extracted value stays within the L1.
+	Sovereignty	Based rollups can still issue a governance token and use it to make decisions about the network
−	No MEV income	Block builders (L1 actors) will not necessarily be motivated to maximize L2 fees
−	Constrained sequencing	Pre-confirmations [20] are hard and the actual ordering of transactions may not be enforced by the L1 actors building the blocks.

Table 1: Advantages (+) and disadvantages (−) of based rollups according to Drake [4], along with a short justification of the trait.

achieved. The process of driving the rollup's sequencing is not precisely defined, and there may be many ways to implement this functionality. Buterin says that in based rollups, "L2 blocks are L1 transactions" [20] and based sequencers indeed work by proposing blocks to an Ethereum smart contract.

It is important to note that based sequencing is less about deriving a sequence of transactions to be ordered than the ability to build blocks. While this distinction may be useless in most blockchain environments, there are situations for which it is important. For example, some blockchains may not wish to allow permissionless block building (or at least not without additional requirements) but might want to have an ordering of transactions supplied by L1 (e.g., those implementing things like sequencer level security [21] or for cross-chain communication). However, the ability to merely order included transactions may be beneficial to that chain: as an argument for fairness or as a requirement for the exact conditions under which a transaction should be analyzed as malicious.

Based rollups are purported to have several advantageous traits that do not introduce additional trust assumptions, though they do not come for free. Table 1 shows the initial traits listed by Drake [4]. These imply others: for a positive example, permissionless L2 block proposal implies censorship resistance on the L2; for a negative example, the design may increase L2 transaction latency. Shortly after this definition was proposed, Buterin suggested that based sequencing may be "total anarchy" [22] as multiple L2 blocks could be proposed within the same L1 block, resulting in wasted gas and computation. This can be overcome by proposer-builder separation (PBS) (see e.g., [23]) for the underlying blockchain, with at most one L2 block per L1 block and delayed validity proofs. We explore the validity of these traits (as well as Buterin's anarchy concerns) in Section 3.2.

3 Case Study: Taiko

As of 6 January 2025, there are only a handful of based rollups in operation, including Taiko [5]. Taiko is a ZK rollup that describes itself as a "based contestable rollup", which we investigate as a case study. Taiko was chosen as it has the largest Total Value Locked (TVL) among based rollups at the time of writing ($295 million USD [3] on 6 January 2025), which indicates its popularity among blockchain users. The data was collected for the initial version of the Taiko based rollup, prior to their "Ontake" upgrade [26], which occurred at Taiko block height 538304 on 7 November 2024. As a result, the data may not accurately reflect the current state of Taiko or based rollups in general. Future based rollups may have different designs, satisfy additional properties, or refine the notion of a based rollup altogether. Nonetheless, this data provides insights into challenges of building based rolups and can help to shape future based rollup designs. In Section 3.1 we describe the architecture of Taiko. Then, in Section 3.2 we evaluate the distribution of L2 block proposers and related based rollup concerns. Finally, we compare Taiko to Scroll – a ZK rollup with a centralized sequencer – in Section 3.3.

3.1 Architecture

The Taiko rollup differs from other rollups in part because of its based design. As with other rollups, some parts of the system are on-chain while others operate off-chain. The system consists of smart contracts, a mempool, an execution engine, and a consensus client.

The smart contracts for Taiko implement the L1 sequencing, canonical bridging, and proof verification functionalities, among others. Taiko uses a smart contract[1] on Ethereum to propose L2 blocks. Bridging functionality implemented by Taiko is similar to that of other rollups. Proof verification is similar, though the rollup supports *tiers* of validity proofs: lower tiers include proofs generated by Trusted Execution Environments (TEEs) (see e.g., [27]), while higher tiers use zero-knowledge proof systems [10] to prove state transitions. Tiered proofs are a Taiko-specific design, and are not a defining feature of based rollups.

Since the submission of all L2 transactions to the L1 contract would negate the cost savings of using an L2, Taiko has an off-chain mempool client. The mempool is in the L2 execution engine, a modified fork of the Go-Ethereum[2] client. Deposit transactions, which bridge assets from L1 to the L2 (i.e., mint or unlock a representation of the L1 asset on L2), must still be submitted to the relevant bridging contract on Ethereum to be processed.

Actors who want to propose a block can run a *proposer*. The proposer service calls `txpool_content` to get pending transactions from the mempool and build blocks. In turn it calls the `proposeBlock` function which emits a `BlockProposed` event. The proposer must submit the parameters for the block, which include things like the proposer's signature and the transaction list. Arguments are passed as `calldata` or blobs [28]: data which is not accessible on the L1 beyond the function call, but is included in the state root for the L1 block that

[1] Proxy: `0x06a9Ab27c7e2255df1815E6CC0168d7755Feb19a`
Implementation: `0xBA1d90BCfA74163bFE09e8eF609b346507D83231`

[2] `https://geth.ethereum.org/`

processes the transaction. Proposed blocks are *soft commitments* to the rollup's state. Taiko does not use EIP-4844 blobs for data availability.

Taiko consensus clients listen for `BlockProposed` events emitted on the L1 and uses these to update the canonical L2 chain. These events create blocks that are merely `proposed`: such a block may still be determined valid or invalid. A proposed block will be invalidated if transactions included within it are not valid (has an invalid signature or nonce, not enough ETH for the fees, or some other common conditions). A blocks status can progress to `proved` if the assigned prover for the block provides a validity proof for the execution of the block (via a call to `proveBlock` which emits a `TransitionProved` event): a proof that, given the previous state root, the state transition function (i.e., Ethereum Virtual Machine execution), and the transaction list, the resulting block is one that is valid. In the current design, the prover submits a bond of tokens alongside the proof.

The Taiko rollup is not simply a based rollup, but a "based contestable rollup" [29]. In this design, a low-tier proof for a block which was previously submitted can be contested by anyone. This means that anyone can claim that the proof is invalid and request that a higher-tier proof is provided (the highest tier proofs cannot be contested). In such a case, a `TransistionContested` event is emitted and a bond is collected. When a higher tier proof is submitted, the honest actor receives a reward. This design borrows from optimistic rollups and is not necessary if all proofs are of the highest tier (zero-knowledge proofs).

Proved blocks may not make it into the final L2 blockchain. Proved blocks are those for which the state transition they outline is valid, but this may be a transition from an invalid L2 state. In particular, an ancestor of the L2 block may have been contested and removed from the L2 chain. Taiko uses the `verified` status to indicate blocks which have been proven and for whom there is a chain of blocks back to the Taiko genesis block. That is, `verified` blocks make up the final L2 chain. Batches of blocks are verified by calling the `verifyBlocks` function on the Taiko smart contract. Any uncontested blocks which are past their contestable period are assumed correct and can be verified by anyone. Verified blocks provide *hard commitments* for the rollup's state.

3.2 Analysis

In this section we report on observed interactions with the Taiko network on Ethereum. We are interested in determining how the system may differ when compared to a standard (non-based) rollup. Our data is collected[3] from the Ethereum blockchain from block 19773965 (May-01-2024 08:03:47 AM +UTC) to 21136529 (Nov-07-2024 03:00:23 PM +UTC), using Infura endpoints. This represents over 6 months of blocks (1362564 blocks) on Ethereum in 2024, starting when the Taiko L1 smart contract was deployed and ending when the relevant events were no longer observed (due to the Taiko Ontake upgrade). The data was collected by iterating over relevant L1 blocks and collecting transactions

[3] Source code: `https://github.com/jgorzny/based-rollups`

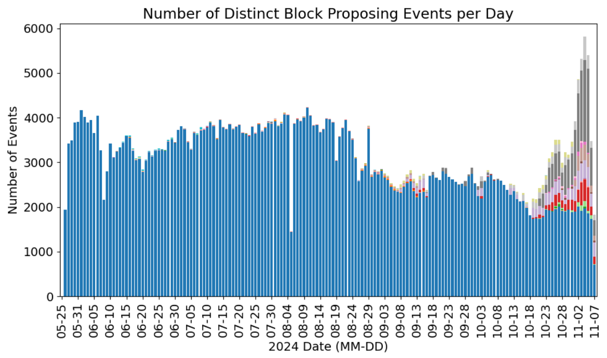
which emitted `BlockProposed`, `TransitionProved`, `TransitionContested`, or `BlockVerified` events. Note that while the data we collect is from the time the Taiko contract is deployed, the first Taiko block is proposed on 25 May 2024. We counted the number of Externally Owned Accounts (EOAs) on Ethereum which made transactions that emitted key events, the frequency of these events, and which parts of the based rollups contribute to the on-chain costs of the network.

The charts in Figure 1 illustrate various findings related to Taiko events. Figure 1a is a stacked bar chart of the number of transactions on each day that emitted a `blockProposed` event. Each color in the chart is a different EOA, though it is clear that the vast majority of transactions originate with a single address (dark blue). This address[4] is the `TaikoBeat` proposer operated by Taiko Labs, and is singlehandledly responsible for over $498145/538303 \approx 92.54\%$ of all block proposing events. There are 100 total addresses which submitted a block proposing transaction during the period of blocks studied. On average 3242.79 block proposing transactions were sent on a day, with the largest number being 5818 proposals in a single day. Figure 1b shows the number of EOAs who either proposed, proved, or verified a block each day in the studied period. On average there were 11.80, 12.02, and 7.15 proposers, provers, and block verifiers respectively. The maximum number of distinct proposing and proving EOAs was 43, with a maximum of 21 verifiers. Finally, Figure 1c shows a scatter plot where the x-axis is L1 block numbers and the y-axis is the number of `BlockProposed` events in that L1 block, if that L1 block contains at least one L2 block proposal. This not only shows that most L1 blocks contain either 1 or 2 Taiko block proposal and it took some time for L1 blocks to frequently contain more than 2 L2 block proposals. The distribution of L2 block proposals in L1 is also shown in Figure 1c. The vast majority (98.69%) of L1 blocks resulted in either 0 or 1 block being proposed.

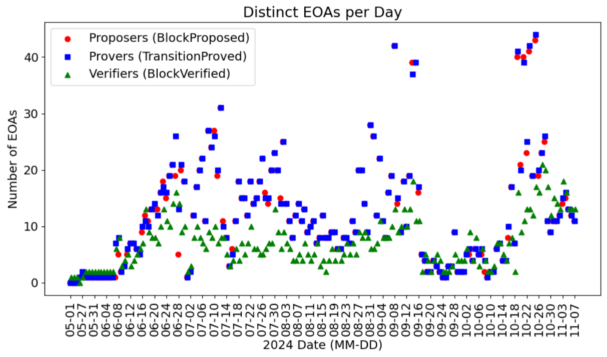
3.3 Comparison

The Taiko based rollup therefore has a different architecture from other non-based rollups. There is an additional requirement of a mempool client and the role of the sequencer is implicit rather than explicit. The notion of submitting blocks (or batches) as soft-commitments from the sequencer is replaced by the requirement to call a `proposeBlock`. This means the call must record some data within a smart contract so that it can be checked later, which is not always the case for other soft-commitments. Other rollups posts L2 blocks as `calldata` (or as blobs) without explicitly writing state – simply calling a view function on a specified contract; this is impossible for Taiko, which needs to record the time it received the proposed block so that it can be eventually finalized. Moreover, a validity proof for an L2 block is no longer sufficient to consider it final on L1 – the block must also be verified. In a ZK rollup, this is handled by building a coordinator service behind the scenes that is responsible for ensuring that proofs are only generated for soft-commitments that are finalized on the L1.

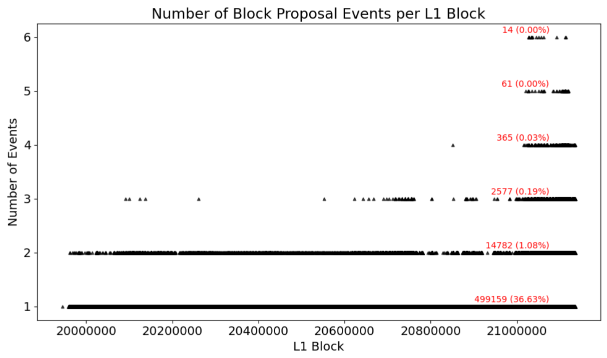
[4] `0x000000633b68f5D8D3a86593ebB815b4663BCBe0`



(a) A stacked bar chart showing the number of `BlockProposed` events emitted per day. Each color is a EOA who sent the transaction that resulted in the event.



(b) A scatter plot showing the number of distinct EOAs that sent a transaction that emitted a key event for Taiko. Recall that only one `TransitionContested` was emitted (by a single EOA).



(c) A scatter plot showing the number of L2 blocks proposed per L1 block containing a block proposal; the red label shows the count for each non-zero y value. 845606 (62.06%) L1 blocks did not propose an L2 block.

Fig. 1: Various illustrations related to block proposal in Taiko.

Rollup	Distinct TX	Duplicate TX	% Duplicate
Taiko	1076696	156318	14.52%
Scroll	308481	0	0.00%

Table 2: Transactions (TX) performing multiple rollup actions at once.

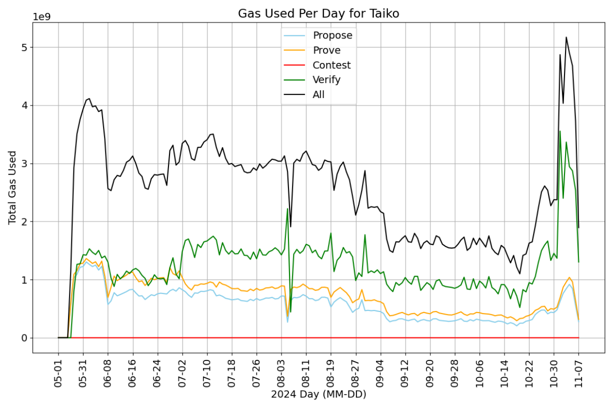
This explicit on-chain verification step may also be more costly in a based rollup and may negatively affect the blockchain performance.

To evaluate this claim, we compare Taiko to Scroll. Scroll is a ZK rollup with a centralized sequencer; it is not a based rollup. Scroll was chosen as it is a ZK rollup which uses the same ZK proof system as Taiko, halo2 [30], in order to make as much of a direct comparison as possible. Scroll proves every block on its network and we describe the process briefly: transactions are put into blocks, which are put into chunks, which are then put into batches which are finally posted on L1. Each of these has an associated validity proof. First, the Scroll network commits a batch of L2 transactions on-chain which emits a `CommitBatch` event. This is similar to Taiko in that a proof is necessary to finalize the batch, which results in a `FinalizeBatch` event, but the batch may not be a single L2 block. The batch consists of part of a *chunk* to be proved, and may therefore contain data from multiple L2 blocks. This allows Scroll to maximize prover utilization (which needs a specific set of memory to prove a block regardless of whether it's full; small blocks can be virtually merged to use all this space). This also allows them to post batches only when necessary, rather than for every block, and may result in multiple blocks being contained in a single batch. Scroll can emit a `RevertBatch` if operator wishes to revert a batch; in the past, 55 batches were reverted to fix a bug in batch compression [31]; this occurred on 3 July 2024 and is the reason for the sharp drop of costs on that day.

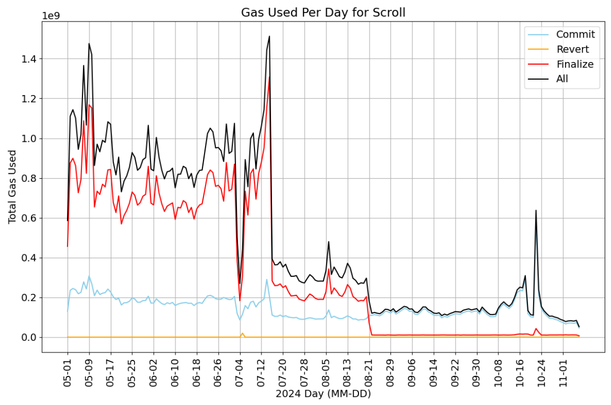
Table 3 displays data regarding the distribution of the events for both Taiko and Scroll. We can see that almost every Taiko block proposed was proven. There is a slight discrepancy that arises from the fact that the final Scroll blocks proposed in the dataset were to be finalized outside of the block range studied. Similarly, block verification happens after blocks are proven, so the verification count is smaller than the proposed count as verification will come after the last block studied. There was a single transition contested event[5] and it is unclear why it occurred; it may have been a deliberate test or a legitimate bug. Taiko proposed 3.12 times as often as Scroll the number of batches that Scroll committed. In terms of blocks, Taiko produced $538,303$ blocks while Scroll produced $5,609,661$ L2 blocks; Taiko produced less than 10% as many as Scroll. Table 2 shows the total number of relevant L1 transactions for each rollup; Taiko had just short of 15% of transactions do more than one action at a time.

Figure 2 breaks down the daily L1 costs of the two rollups. Figure 2a shows the costs in units of gas times 10^9 for transactions that emitted key based se-

[5] Transaction: `7600471694620e19c3296a4e26fc753149cbcd9803f37747521aa3399261ced8`



(a) A breakdown of daily L1 gas costs associated with Taiko.



(b) A breakdown of daily L1 gas costs associated with Scroll.

Fig. 2: On-chain transaction costs for Taiko and Scroll.

quencing events. Note that the spike near 31 October 2024 may be explained[6] by a contract upgrade ahead of the Ontake fork. Verification of proofs is the most expensive component, and block proposal and proving are similar in cost. Costs related to contested transactions are almost entirely zero given the infrequency of the event. Compared with the costs for Scroll in Figure 2b, Taiko is nearly 5 times more expensive: the units on the y-axis is the same in both charts, but Scroll's peak daily amount is around $1.5 \cdot 10^9$ while Taiko's is over $5 \cdot 10^9$. The difference is also visualized in Figure 3a, which shows the cumulative cost in units of gas required to operate each network for the studied block duration (this time in gas units times 10^{11}). From this figure, it's clear that Taiko's operating costs are worse than Scroll's even before the spike in daily costs near the end of

[6] `https://x.com/taikoxyz/status/1849703188678705365`

Taiko Event	Count	Scroll Event	Count
`BlockProposed`	538303	`CommitBatch`	172120
`TransitionContested`	1	`RevertBatch`	55
`TransitionProved`	538304	`FinalizeBatch`	136360
`BlockVerified`	53806		

Table 3: Event counts for the studied period. Note that for Taiko, one more transition is proved than blocks are proposed, as the genesis block did not emit a `BlockProposed` event.

the studied period. Moreover, this is true despite the fact that Scroll commited batches between the time the Taiko contract was deployed and the first Taiko block was proposed (about 27 days). These costs might not be accurate: these transactions may perform other actions as well as calling the appropriate rollup functions, though calling non-rollup functions would likely not be desirable.

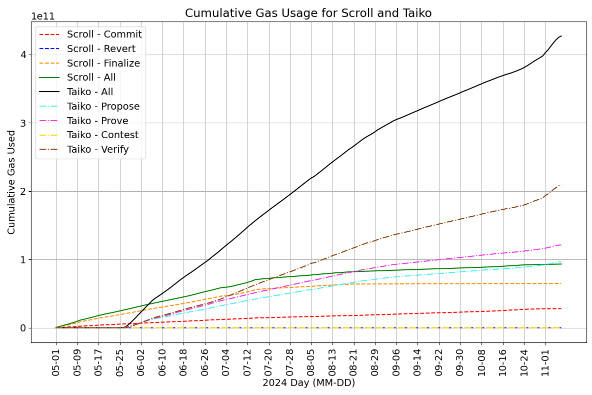
Figures 3b and 3c show the differences in block generation time for each chain. Each point is the average block generation time computed over the last 500 L2 blocks, starting with the first 500 during the studied period; units are in seconds. The red line on each is the average over the last 10 points (5000 blocks). Taiko's block generation time is much more varied than Scroll's, and Taiko blocks are often 20 or 30 seconds apart, though they are closer together near the end of the studied period. On the other hand, Scroll bocks are typically 3 seconds apart, and sometimes much less than that. Scroll's worst average block time is still better than Taiko's best average block time.

4 Discussion

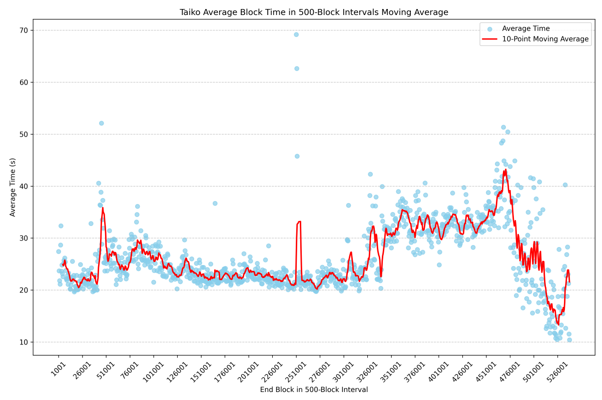
In this section, we discuss the comparison between Taiko and Scroll and aim to to evaluate the claimed positive traits of based rollups (c.f. Table 1).

Performance. The results of Section 3.3 show that Taiko was nearly five times more expensive to operate and nearly ten times slower to produce blocks. The expense may come from the multiple steps required to prove blocks, but is more easily explained by the lack of batching and aggregation, as well as any additionally advantageous compression. It is not surprising that if most L1 blocks that proposed Taiko blocks only proposed one L2 block, the performance gains would not be as great as if batching transactions occurred. Moreover, this explains the delays in block generation as well. In short, based sequencing requires additional effort to be competitive with those which do not use the L1 to build blocks. It is worth noting that the Ontake update to Taiko did introduce batching to the system, but given the downsides of omitting this functionality, it might be worth considering that feature as essential for based rollups.

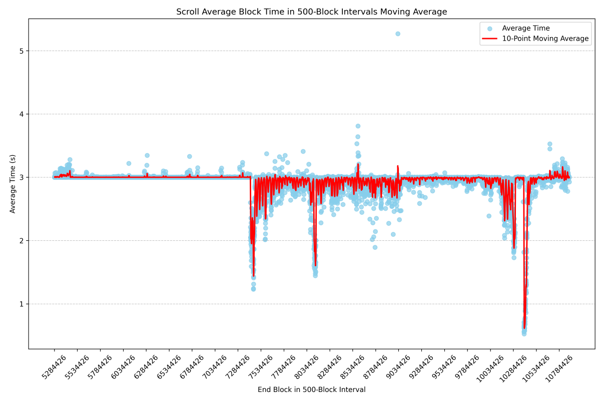
Liveness. The Taiko rollup appears to suggest that the liveness trait is valid. Although a single proposer is responsible for the majority of the L2 blocks proposed (c.f. Figure 1a), there are other block proposers who could continue to



(a) Cumulative L1 gas costs for Taiko and Scroll.



(b) Taiko's average blocktime.



(c) Scroll's average blocktime.

Fig. 3: Cumulative gas costs for both networks (a) and average blocktimes for Taiko (b) and Scroll (c).

progress the L2 blockchain in the event that the `TaikoBeat` proposer is offline. However, the Taiko team still controls the ability to update the contracts which negatively affect this trait; barring any such issues, the data confirms that based rollups have strong liveness.

The Taiko chain does not have "total chaos" from too many L2 blocks proposed in a single L1 blocks; it was designed in such a way that this is not problematic. Although some L1 blocks had two or more L2 blocks proposed in them, the majority (98.69%) of L1 blocks did not propose any L2 block or proposed exactly one. In our investigation, no two L2 blocks proposed at the same block had the same L2 block number, so there appears to be no wasted effort. However, future work is necessary to check if two L2 blocks within the same L1 block actually shared transactions – this might still mean wasted blockspace and is not immediately apparent from the event log. It will also be important to revisit this concern for based rollups that allow batches of blocks to be posted in a single L1, as these batches increase the likelihood of wasted effort.

Decentralization. The data in Figure 1b shows that based rollups have the potential to be decentralized. Although there is a number of proposers, they do not propose the majority of L2 blocks. This is similar to L1 blockchains with mining pools: a large number of users participate in the network but block production is nonetheless centralized to a smaller number of parties [32]. Unlike in these L1 networks, this may change more easily for the Taiko network. Based rollups do not necessitate a Proof-of-Stake (PoS) barrier to entry as no funds beyond those necessary to transact and the appropriate proving hardware are required. Taiko's current design includes an indirect PoS-like mechanic as provers must be bonded, though this may not be necessary when only the highest tier proofs are used. The current bond may be too expensive to attract additional network participants. Nonetheless, the trait appears plausible in mature rollups.

Simplicity. The engineering effort of a based rollup remains difficult to judge. Just like Ethereum itself, Taiko uses a consensus client and an execution client, and has smart contracts as well. The development and maintenance of these systems is non-trivial and even without the need for specific features like escape hatches, care is needed to ensure all requirements are implemented and can be evaluated [33]. This is made worse by the introduction of cryptoeconomic incentives which need to be carefully balanced, the choice of proof systems, and the inability to optimize particular functionality. It is unclear if based rollups are truly simpler than standard designs; such a verdict will need to be rendered only after there are mature feature-complete examples of both kinds of rollups.

L1 Economic Alignment & Sovereignty. Taiko appears to be aligned with Ethereum while maintaining sovereignty. The design of the rollup necessitates several Ethereum transactions to build L2 blocks, so those participating in Taiko's block building process must operate on and within the Ethereum ecosystem. Additionally, Taiko has a token (TKO) which it uses as a bond for block provers and governance; this will provide sovereignty and further aligns its participants with the Ethereum blockchain economically. These traits appear plausible, but a more rigorous definition of each and further study is necessary. It may be too early to determine the usefulness of the TKO token within the Taiko ecosystem.

4.1 Threats to Validity and Limitations

This work provides an initial comparison with Taiko and Scroll; however, several potential threats to validity must be considered. First, the comparison is not a direct one as there may be additional architectural differences between these rollups aside from their sequencer (as is evident by tiered proofs). Second, the data requires deeper analysis to ensure that duplicate transactions do not have their costs counted twice; ideally, a function-level analysis of gas cost would be used. Third, Taiko's subsequent upgrades and other based rollups may not suffer from the same problems as the version of Taiko studied. Future developers of based rollups should repeat this study with the most modern, optimized versions of these systems that exist. Critically, the Taiko Ontake update added critical features like block batching [26], which could drastically affect operating costs and is already present in Scroll. Finally, even if cost metrics can be directly compared, there are other factors – such as transaction latency – that may be omitted by such an analysis but important when choosing a rollup design.

5 Conclusion

In this work, we discussed the expected advantages and disadvantages of based rollup design before reporting on the design of a real-world based rollup, Taiko. Using Taiko as an example based rollup, we confirmed that there are few technological challenges prohibiting their development. However, there is work necessary to make them cost-effective, there is room to improve on the design, and these rollups appear decentralized but are largely operated by a single entity. Nonetheless, we believe based rollups are a feasible and promising layer two solution that will improve over time. Without optimization for features that are key to centralized sequencer-based designed, based rollups do appear likely to be competitive.

There are a number of interesting directions for future work. First, a more fine grained-evaluation of Taiko is possible: e.g., measure the finality and proof delay time, evaluate other (based) rollups, include the costs for data availability, or measure per-function gas usage. Second, there are likely a number of interesting questions about Taiko itself: e.g., if transitions may need to be contested, how many contesting actors are active over a given time period? Third, one could revisit the claimed properties of based rollups: e.g., are L1 actors really apathetic to maximizing based rollup block fees? A fine-grained evaluation of their purported simplicity and economic alignment is omitted from this work and should be conducted. As Buterin [34] suggests that rollups are a cultural extension of Ethereum, it is important that we understand and improve them.

Acknowledgements. The authors would like to thank the anonymous reviewers of CAAW 2025 for their numerous helpful and insightful comments.

References

1. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger, 2014. Ethereum Project Yellow Paper, `https://ethereum.github.io/yellowpaper/paper.pdf`.
2. Lewis Gudgeon, Pedro Moreno-Sanchez, Stefanie Roos, Patrick McCorry, and Arthur Gervais. SoK: Layer-two blockchain protocols. In *Financial Cryptography*, volume 12059 of *Lecture Notes in Computer Science*, pages 201–226. Springer, 2020.
3. Bartek Kiepuszewski. L2Beat value secured, 2025. `https://l2beat.com/scaling/tvs`.
4. Justin Drake. Based rollups—superpowers from L1 sequencing, March 2023. `ethresear.ch/t/based-rollups-superpowers-from-l1-sequencing/15016/6`. Accessed 2 Feb. 2024.
5. Taiko. `taiko.xyz/`. Accessed 2 Feb. 2024.
6. Scroll. `https://scroll.io/`. Accessed 12 Dec. 2024.
7. Rami Khalil, Alexei Zamyatin, Guillaume Felley, Pedro Moreno-Sanchez, and Arthur Gervais. Commit-chains: Secure, scalable off-chain payments. Cryptology ePrint Archive, Paper 2018/642, 2018. `https://eprint.iacr.org/2018/642`.
8. Patrick McCorry, Chris Buckland, Bennet Yee, and Dawn Song. SoK: Validating bridges as a scaling solution for blockchains. Cryptology ePrint Archive, Paper 2021/1589, 2021. `eprint.iacr.org/2021/1589`.
9. Louis Tremblay Thibault, Tom Sarry, and Abdelhakim Senhaji Hafid. Blockchain scaling using rollups: A comprehensive survey. *IEEE Access*, 10:93039–93054, 2022.
10. Jan Gorzny and Martin Derka. A rollup comparison framework. *CoRR*, abs/2404.16150, 2024.
11. Shashank Motepalli, Luciano Freitas, and Benjamin Livshits. SoK: Decentralized sequencers for rollups, 2023.
12. Akaki Mamageishvili and Jan Christoph Schlegel. Shared sequencing and latency competition as a noisy contest. *CoRR*, abs/2310.02390, 2023.
13. Akaki Mamageishvili and Edward W. Felten. Incentive schemes for rollup validators. In *MARBLE*, Lecture Notes in Operations Research, pages 48–61. Springer, 2023.
14. Emanuel Onica and Marius Georgica. Can smart contracts become smart?: An overview of transaction impact on Ethereum DApp engineering. In *DICG@Middleware*, pages 31–36. ACM, 2023.
15. Stefanos Chaliasos, Itamar Reif, Adrià Torralba-Agell, Jens Ernstberger, Assimakis Kattis, and Benjamin Livshits. Analyzing and benchmarking ZK-rollups. *IACR Cryptol. ePrint Arch.*, page 889, 2024.
16. Roshan Palakkal, Jan Gorzny, and Martin Derka. SoK: Compression in rollups. In *ICBC*, pages 712–728. IEEE, 2024.
17. Akaki Mamageishvili and Edward W. Felten. Efficient rollup batch posting strategy on base layer. In *FC Workshops*, volume 13953 of *Lecture Notes in Computer Science*, pages 355–366. Springer, 2023.
18. Vitalik Buterin, Dankrad Feist, Diederik Loerakker, George Kadianakis, Matt Garnett, Mofi Taiwo, and Ansgar Dietrichs. EIP-4844: Shard blob transactions, 2022. `https://eips.ethereum.org/EIPS/eip-4844`.
19. Davide Crapis, Edward W. Felten, and Akaki Mamageishvili. EIP-4844 economics and rollup strategies. *CoRR*, abs/2310.01155, 2023.

20. Vitalik Buterin. Epochs and slots all the way down: ways to give ethereum users faster transaction confirmation times, June 2024. `https://vitalik.eth.limo/general/2024/06/30/epochslot.html`. Accessed 1 Jan. 2025.
21. Martin Derka, Jan Gorzny, Diego Siqueira, Donato Pellegrino, Marius Guggenmos, and Zhiyang Chen. Sequencer level security. *CoRR*, abs/2405.01819, 2024.
22. Vitalik Buterin. Epochs and slots all the way down: ways to give Ethereum users faster transaction confirmation times, January 2021. `https://vitalik.eth.limo/general/2021/01/05/rollup.html`.
23. Lioba Heimbach, Lucianna Kiffer, Christof Ferreira Torres, and Roger Wattenhofer. Ethereum's proposer-builder separation: Promises and realities. In *IMC*, pages 406–420. ACM, 2023.
24. Jan Gorzny, Po-An Lin, and Martin Derka. Ideal properties of rollup escape hatches. In *DICG@Middleware*, pages 7–12. ACM, 2022.
25. Philip Daian, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In *SP*, pages 910–927. IEEE, 2020.
26. Taiko mainnet upgrade and Ontake hardfork. `https://taiko.mirror.xyz/OJA4SwCqHjF32Zz0GkNJvnHWlsRYzdJ6hcO9FXVOpLs`. Accessed 17 Jan. 2025.
27. Xiaoqing Wen, Quanbi Feng, Jianyu Niu, Yinqian Zhang, and Chen Feng. TeeRollup: Efficient rollup design using heterogeneous TEE. *CoRR*, abs/2409.14647, 2024.
28. Suhyeon Lee. 180 days after eip-4844: Will blob sharing solve dilemma for small rollups?, 2024.
29. Based booster rollup (BBR): A new major milestone in Taiko's roadmap, December 2023. `https://taiko.mirror.xyz/Z4I5ZhreGkyfdaL5I9P0Rj0DNX4zaWFmcws-0CVMJ2A`. Accessed 1 Oct. 2024.
30. Electric Coin Company. halo2: A zero-knowledge proof system. `https://zcash.github.io/halo2/`. Accessed: 6 Jan. 2025.
31. Incident report for scroll. `https://status.scroll.io/incidents/44k6s4qg6kcs`. Accessed 17 Jan. 2025.
32. Chao Li and Balaji Palanisamy. Comparison of decentralization in DPoS and PoW blockchains. In *ICBC*, volume 12404 of *Lecture Notes in Computer Science*, pages 18–32. Springer, 2020.
33. Jan Gorzny and Martin Derka. Requirements engineering challenges for blockchain rollups. In *RE Workshops*, pages 340–347. IEEE, 2024.
34. Vitalik Buterin. Layer 2s as cultural extensions of Ethereum, May 2024. `https://vitalik.eth.limo/general/2024/05/29/l2culture.html`. Accessed 6 Jan. 2025.