ZK-Rollups on the Wild:
A Public Dataset for the Polygon zkEVM

Abstract

Permisionless blockchains provide transparency, but converting raw on-chain traces into research-ready assets still demands specialized infrastructure, highly specialized low-level technical expertise, and non-trivial cost. These frictions are especially evident in Layer 2 environments (where Polygon zkEVM is an example), due to its added complexity on top of an already complex system. To lower this barrier, we assembled and released the corpus of Polygon zkEVM activity from its genesis up until 2024, extracted from a synced node and indexed into analysis-friendly tables. In this work, we describe the collection and transformation pipeline carried out over the Polygon zkEVM data, and showcase a representative analysis on user behavior patterns. We conclude with open questions and paths for future work. Moreover, we publish our code, notebooks and dataset to promote reproducibility, independent access to the data, and community contributions.

Keywords: Layer 2, blockchain, ZK-Rollups, public dataset.

1. Introduction

The permissionless blockchain ecosystem is fundamentally built on principles of decentralization and transparency. However, the complexity and opacity of accessing blockchain data present significant challenges for both end users and researchers, particularly when it comes to Layer 2 (L2) [1] solutions.

Currently, the way to obtain blockchain data is through the deployment of an archival node [2, 3], that is, a node that stores all past blockchain history up to the present. For Ethereum, the process is rather opaque and demands significant computational resources. The official documentation specifies minimum requirements of a 4-core CPU, 16 GB of RAM, and a 2 TB NVMe SSD merely to maintain synchronization with the chain [4]. Synchronizing an archival node from the genesis block is even more demanding, as it requires processing and validating several years of historical blockchain data to bring the node fully up to date. Thus, this process is, arguably, not practical to enable any individual to deploy its own solution to obtain data. The challenge becomes even greater in the context of syncing a node for a ZK-Rollup, since the relatively novel technology is prone to breaking, non-backward compatible changes during its lifecycle, making the procedure even harder and less accessible. Once the node is synced, users can obtain blockchain data through the use of Remote Procedure Calls (RPCs) [5]. While this is a valid way to gather data, it is a highly technical approach to obtain it, making it unfeasible and challenging for a number of non-technical users. On top of this, it is usually a costly and slow way to obtain these data, since those petitions add a fair amount of overhead to the overall system.

Instead, users can decide to rely on third parties to gather this data, such as Etherscan [6], Arbiscan [7], zkEVMPolygonscan [8], among others. Although they are a convenient way to obtain blockchain data, it has been shown that this approach entails significant economic cost in the long run, as well as usually falling short of meeting the expectations and needs of final users. Additionally, we believe that *independent* access to Layer 2 should be granted since, while performing this study, we have found (and reported) inconsistencies between our data and other providers, such as zkEVMPolygonscan [9].

We are strongly convinced that anyone requiring blockchain data should be able to obtain it in the most easy and direct way, without dealing with complicated infrastructure or hardware. The access to these data is crucial for research purposes, like analyzing Maximum Extractable Value (MEV) [10], Airdrop designs [11], Automated Market Maker (AMM) [12, 13], or to study the viability of the adoption and interaction among several Layer 2 solutions.

This paper presents a complete dataset, accompanying code, and an initial exploratory analysis of transaction-level data from Polygon zkEVM. Our data comprises the interval between blocks 0 and 18.799.951, covering the dates between 24/03/2023 (its genesis block) and 31/12/2024. A full description of the code and data can be found in Section 3.

1.1. Why Polygon's zkEVM Rollup?

In recent years, permissionless blockchains like Bitcoin or Ethereum have suffered from scalability issues [14]. To solve this issue in the case of Ethereum, the ecosystem is moving towards a rollup-centric roadmap [15], where ZK-Rollups have a key role to achieve scalability while maintaining security and decentralization.

Given the important role of L2 scalability solutions and, in particular, ZK-Rollups, we strongly believe that this is an opportunity for researchers and users to conduct experiments, analyze, and explore the L2 space and, in particular, ZK-Rollups.

Polygon's zkEVM [8] represents a significant advancement in this space, offering a scalable and efficient Ehtereumcompatible environment through the use of Zero-Knowledge Proofs (ZKP) [16]. Started in March 2023 on Mainnet, Polygon zkEVM is an L2 scaling solution for the Ethereum blockchain that relies on validity proofs (ZKP) to reduce the overhead of transacting and interacting on the blockchain. Moreover, Polygon zkEVM is a Type 2 zkEVM [17], ensuring compatibility with existing Ethereum Smart Contracts. As of December 2024, Polygon zkEVM is among the top 10 ZKP-based chains with a Total Value Locked (TVL) of around $ 85M [18].

For this reason, making our Polygon zkEVM data available, as well as the code used to analyze the data, will encourage and enable further research in this direction, expanding our understanding and improving ZK-Rollups.

1.2. Our contributions

This work makes three primary contributions:

Public –independent– dataset. We introduce and publicly release a structured, ready-to-use dataset containing all existing transaction-level data from Polygon's zkEVM. This dataset is designed to be easily readable and immediately accessible, eliminating the need for users to synchronize a node or manually parse raw blockchain data.

Open-source codebase. We release all code necessary to replicate the dataset construction from a synchronized archive node, ensuring transparency, reproducibility, and extensibility for future work.

Data exploration. We provide a thorough in-depth study of the data gathered, contributing with both simpler and more complex metrics regarding the pattern usage of the Polygon zkEVM.

1.3. Paper organization

The rest of this paper is organized as follows: Section 3 gives background about how Polygon's zkEVM works, together with the explanation of the architecture used to perform this study. Section 4 presents graphs that analyze several aspects about this network, such as daily metrics, usage pattern metrics, metrics on the size of the data availability posted on-chain, or metrics regarding the bundle of transactions into blocks and batches. Finally, in Section 5 we present the conclusions and the future work proposed for this study.

2. Background and state-of-the-art

Blockchain technology establishes a decentralized ledger in which transactions are recorded in chronologically linked blocks. Each block stores a batch of verified transactions, a timestamp, and the cryptographic hash of the preceding block, thereby ensuring historical immutability. When a transaction is initiated, it is propagated through a distributed network of nodes that collectively verify its validity. Transactions are incorporated into a new block which is appended to the chain, after consensus is reached, a process that makes tampering with past records computationally infeasible without commandeering a majority of the network.

Despite these robust guaranties, blockchains remain fundamentally constrained in their ability to process large volumes of transactions efficiently. Prominent platforms such as Bitcoin and Ethereum typically maintain throughput levels on the order of 10–15 transactions per second, with transaction fees escalating sharply during periods of intense network congestion. To address these scalability challenges, Layer 2 (L2) protocols have emerged as a core area of innovation. These solutions offload transaction execution from the base layer by batching or aggregating transactions on a secondary protocol. Periodically, a succinct proof or summary is committed back to the underlying chain, thereby increasing throughput and reducing fees while largely retaining base layer security. The significance of these advances is underscored by the collective Total Value Locked (TVL) in L2 protocols, which now exceeds $ 40 billion [18].

Among Layer 2 designs, Zero-Knowledge (ZK) rollups represent a particularly impactful development. Here, a dedicated off-chain prover compiles many transactions into a single state transition, computes the updated state root, and generates a compact zero-knowledge proof (often in the form of a zk-SNARK [19]) attesting to the validity of the entire batch [20]. The main chain, rather than processing each transaction, merely verifies this proof and updates its state accordingly. As a result, ZK rollups achieve high throughput, low transaction costs, and fast finality, all while preserving the stringent security guaranties of the underlying blockchain `Cite redacted for submission`.

2.1. L2 to L1 communication

Let us explain how the communication between L1 and L2 occurs, focusing on the particular case of communicating between Ethereum as L1 and Polygon zkEVM as L2.

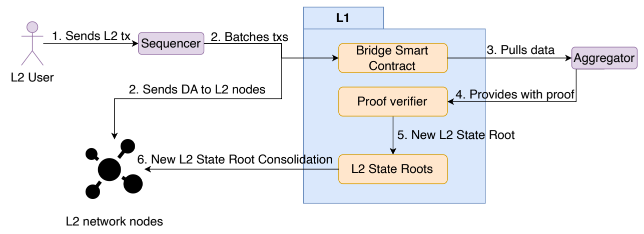

Figure 1: Diagram representing the transaction flow of an L2 transaction.

Figure 1 illustrates the step-by-step that a transaction submitted to Polygon zkEVM follows from the moment that an L2 User submits it, until it is verified on L1 and it is properly updated on L2.

1. **The User sends L2 tx:** the L2 User sends the desired transaction to an L2 Sequencer.

2. **The Sequencer batches txs (and sends Data Availability to L2 nodes):** the Sequencer orders the transactions from the mempool and bundles them in batches to be sent to

both other L2 nodes, and to the L1 bridge smart contract in order to provide with Data Availability.

3. **The Aggregator pulls data:** once the transactions are provided as DA on L1, the Aggregator (also known as the Prover) pulls the data that need to be processed and proved to L1.

4. **The Aggregator provides with the proof:** once the Aggregator has the proof done, it is sent to the L1 smart contract to be verified.

5. **A new L2 State Root is created:** if the proof is correctly validated, the prover smart contract yields a new validated L2 State Root.

6. **The new L2 State Root is consolidated:** this new validated L2 State Root is shared among all peers on the L2 network, so the nodes can continue the process from this new state root.

2.2. L2 data analysis

In this Section, we present previous work that has done a curation and data analysis on other L2s.

The most prominent example of an L2 data analysis study is the dataset curated by Silva et al. [21], where they presented and released a dataset for the ZKSync Era ZK-Rollup, with a special focus on the economic aspect of the network (detailing swap transactions, among others). Following this trend, several studies have extended the public dataset approaches to other Zero-Knowledge rollups and payment network ecosystems. For ZK-Rollups, there exist a number of studies that tackle the topic of data analysis over L2 blockchain systems. Kaczyński et al. [22] applied unsupervised anomaly detection on a ZkSync data dump to identify suspicious transactions and potential DDoS attacks, while Ferreira Torres et al. [23] investigated the extraction of MEV across Layer-2 rollups including zkSync, Arbitrum, and Optimism. In the domain of off-chain payment networks, Feichtinger et al. [24] benchmarked graph neural network architectures using the public Lightning Network channel graph, and the Bitcoin Lightning Network Stats Dataset [25] provides comprehensive network statistics on channels, capacity, and node distribution to allow longitudinal and topological analyzes.

We share their values and vision and, following this trend, we make our contribution to this field providing a dataset to explore the Polygon zkEVM as a ZK-Rollup which is, to the best of our knowledge, the first work to do so.

3. Data description and methodology

In this Section, we present a description of the data we have collected and explain the methodology in this work used obtain these data.

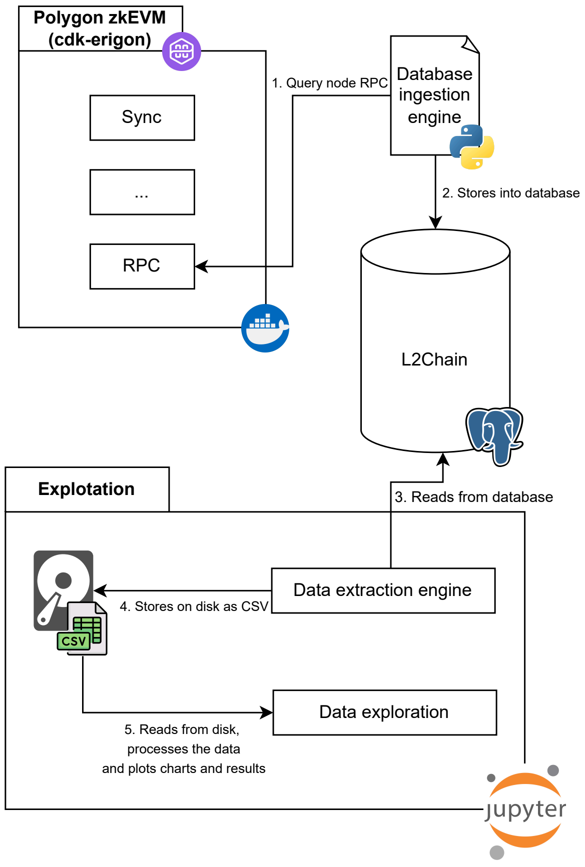

Figure 2: Architecture of our solution.

3.1. Data description

Figure 3 shows a summary diagram of Polygon's zkEVM data architecture. In general, Polygon's zkEVM considers four different types of data structure:

Transactions contains the information that users are sending to the network when interacting with it.

Blocks bundle transactions. A block can be empty, i.e., it may contain 0 transactions.

Batches bundle blocks. A batch can be empty, i.e., it may contain 0 blocks.

Sequences aggregate batches and send them to L1 to comply with the Data Availability requirements.

For the scope of this study, we are interested in the first three aforementioned data structures: transactions, blocks, and batches. In this study, sequencer are not considered, since there are no explicit L2 data about them.

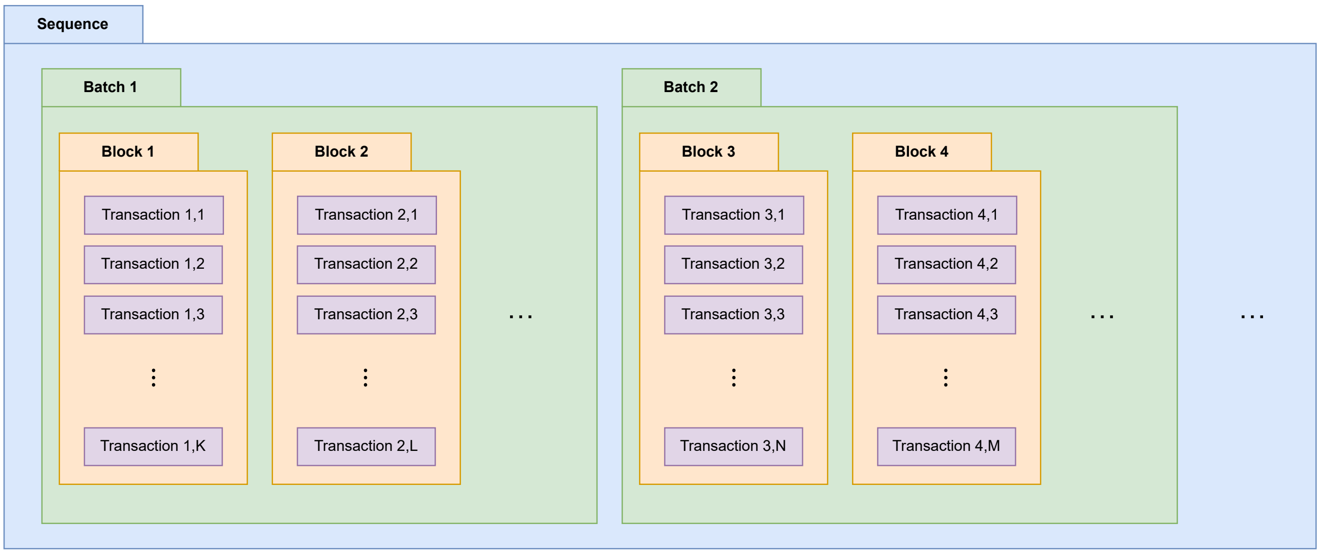

Figure 3: Polygon zkEVM architecture regarding Sequences, Batches, Blocks, and Transactions.

3.2. *Data gathering methodology*

Regarding methodology, we gathered the dataset from a deployment of a Polygon zkEVM node that synchronizes the network [26]. We have processed and stored the data obtained from our synced node into a PostgreSQL database and, after that, we have queried and stored these data on a disk analyze it.

Figure 2 summarizes the architecture of our solution, made up of three independent parts.

1. **Node synchronization and RPC module.** This part of the solution is formed by a docker compose file that spawns a `cdk-erigon` node that synchronizes the Polygon zkEVM network. The RPC module is written in Python [27] and interacts with the RPC server that the node provides to obtain the relevant information for this study.

2. **Creation and handling of our database.** We have created a Python script that creates a PostgreSQL database (previously spawned through a docker compose), reads the data from the node RPC, and stores it on our database. The Entity-Relationship Diagram of the database can be found in Figure 4. In particular, the RPC provides 4 different kind of data: `batches`, `blocks`, `transactions`, and `receipts`.[1]

3. **Ingestion and exploration of the database.** Finally, we have several Jupyter Notebooks that read the data from our database, and store them on disk (as CSV files) for convenience. We load these stored data on memory (using PolaRS [28]), pre-process it, and present aggregated results in the form of graphics and plots (using Vega-Altair [29]) `Cite to the code redacted for submission. The code will be publicly available once accepted.`

[1]Receipts contain additional information about transactions, with which we have merged this information together with the transaction data.

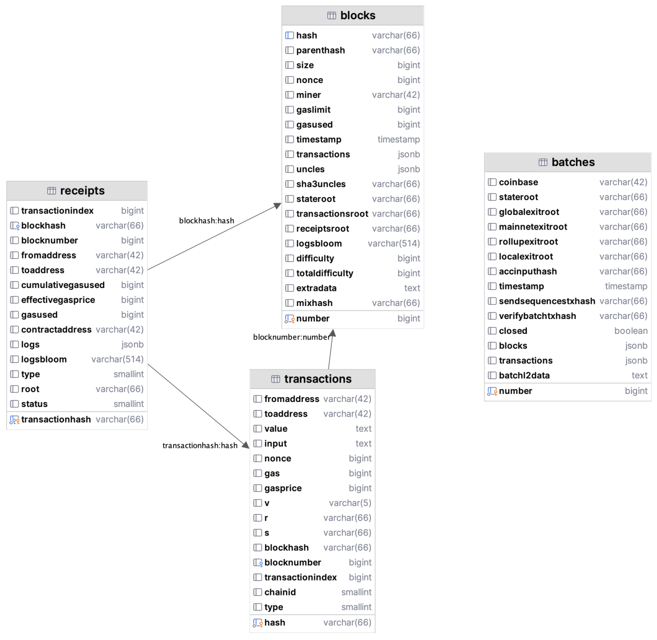

Figure 4: Entity-Relationship Diagram of our database.

4. Analyzing the data

In this Section, we present a detailed data exploration on the dataset described in Section 3.

The dataset provided in this study covers the period between March 24th, 2023 (block number 0) and December 31st, 2024 (block number 18.799.950) [30]. In this period, the network processed 14.557.133 transactions, 18.799.951 blocks, and 2.145.410 batches. All transactions were processed by 709.729 unique users. During this period, 934 contracts were deployed. Table 1 summarizes our dataset.

Start Date	2023-03-24
End Date	2024-12-31
# of Transactions	14.557.133
# of Blocks	18.799.951
# of Batches	2.145.410
# of Contracts	934
# of Unique From Addresses	605.128
# of Unique To Addresses	448.042
# of Unique Addresses	709.729

Table 1: Description of our Polygon zkEVM dataset.

4.1. Daily metrics

In this Section, we present aggregated metrics, grouped by day. In particular, we show data regarding daily count of transactions, blocks, and batches (Figure 5). It is worth noticing that, since the genesis block up until mid February 2024, the transactions line (in blue) and the blocks line (in green) overlap. This is because, during this period, the network functioned in a manner whereby each block corresponded to precisely one transaction.[2] After that period, we can see a more organic and natural behavior on the network.

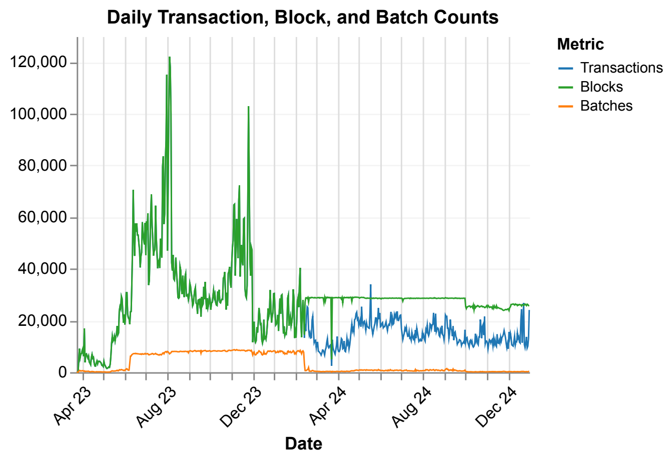

Figure 5: Daily count of transactions, blocks and batches.

Furthermore, Figure 6 shows the mean and total daily block size placed on the network. We can observe that the total amount of daily block size strongly correlates (Pearson correlation of $\rho = 0.944$) with the daily transaction count (Appendix 7, Figure 20), indicating that most of the network usage comes from “simple" transactions (e.g. payments), rather than heavy and large deployments of smart contracts.

[2]The official documentation states the following: “*In the current design, a single transaction is equivalent to one block. This design strategy not only improves RPC and P2P communication between nodes, but also enhances compatibility with existing tooling and facilitates fast finality in L2. It also simplifies the process of locating user transactions.*", but this design choice comes with drawbacks like bloated database and incompatibilities with some dApps [31, 32].

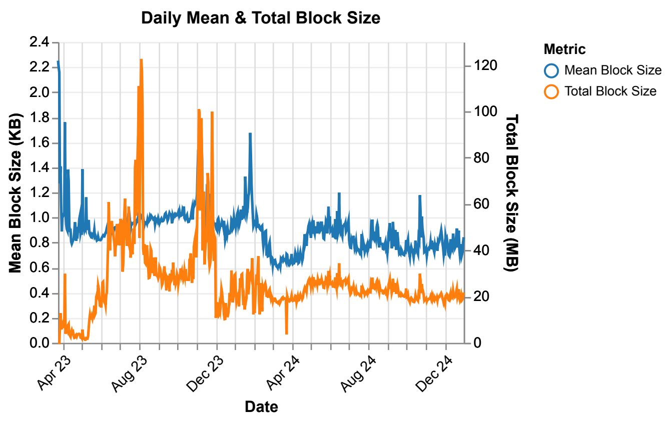

Figure 6: Daily mean and sum of block size

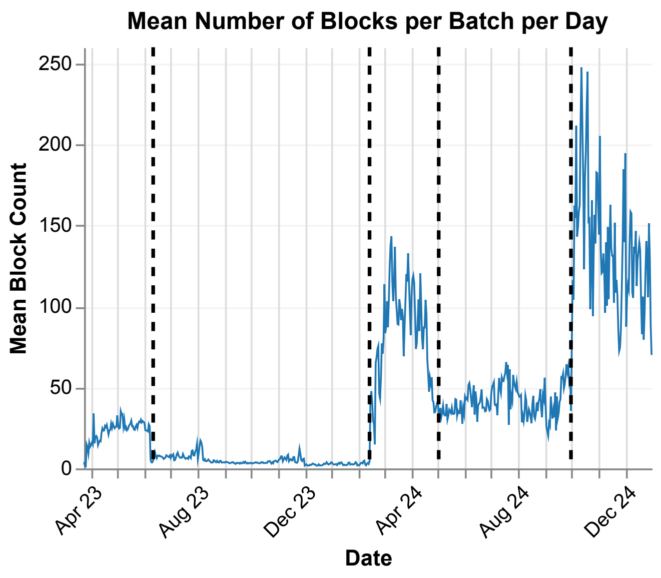

Figure 7: Mean Blocks per Batch per Day

Finally, Figure 7 shows the mean number of blocks per batch processed per day. In the Figure, we can clearly see five different regions that we outlined by the dashed vertical lines on the chart. To the best of our knowledge, the change in the trends in the creation of the blocks is due to the following reasons:

- **Region 1: Mainnet Beta Launch (March-July 2023)**: Polygon zkEVM mainnet beta launched on March 27, 2023, with low but steady block production (≈30 blocks/day). This phase focused on cautious rollout, security, and Ethereum compatibility. The network gained early adoption with over 5,000 smart contracts deployed and Vitalik Buterin processing the first transaction [33], marking strong ecosystem support.
- **Region 2: Network Optimization (August 2023-February 2024)**:

Block production dropped significantly to minimal levels (≈5-10 blocks/day) during infrastructure upgrades. Critical updates included the Dragonfruit [34] and Incaberry [35] hardforks, which improved cryptographic performance, bug fixes, and opcode support. This period prioritized stability and optimization over throughput.

- **Region 3: Activity Surge (March-June 2024)**:
Following a major 10-hour outage in March 2024, the network saw a sharp increase in block production (≈100-150 blocks/day). Recovery involved sequencer improvements and the Feijoa upgrade, implementing EIP-4844 [36] blob support. Proof generation became more efficient, with batches processed faster, coinciding with rapid growth in user activity.

- **Region 4: Stabilization (July-November 2024)**:
Block production stabilized to a moderate rate (≈30-60 blocks/day) during a mature infrastructure phase. Economic optimizations favored larger batch sizes with fewer batches. The network maintained low transaction fees and full developer tooling support, emphasizing cost efficiency and robustness.

- **Region 5: Eggfruit Upgrade Era (December 2024-Present)**:
The Eggfruit [37] upgrade launched on September 30, 2024, introducing the CDK-Erigon [38] sequencer and dramatically increasing block production (≈200-250 blocks/day). This transition improved throughput, continuous scaling capability, and overall network performance, reflecting Polygon zkEVM's technical maturity and readiness for production-scale operations.

4.2. *Usage metrics*

This Section presents a number of usage metrics figures regarding `from` and `to` addresses, and also `function calls`.

Figures 8 and 9 show the 25 most popular `from` addresses, and a cumulative distribution of the number of transactions per `from` address. Similarly, Figures 10 and 11 present the same results with respect to the `to` address.

We can clearly observe that this network is used by a relatively small number of users that account for a large amount of the total number of transactions placed. Specifically, we can see that addresses with 20 or less interactions with the L2 account for 80 % of total transactions, and addresses that appear as `to` in a transaction 5 times or less, account for more than 90 % of the total number of transactions placed. Analysis of the transaction distribution reveals significant concentration in the network. Figure 9 shows that approximately 97 % of transactions originate from the top 100 addresses, while Figure 11 demonstrates that roughly 95 % of transactions are directed to the top 10 receiving addresses.

Moreover, we have computed the intersection between the top 25 from addresses and top 25 to addresses, and we found that the intersection of those sets is 2, indicating a limited overlap between the most active senders and recipients.

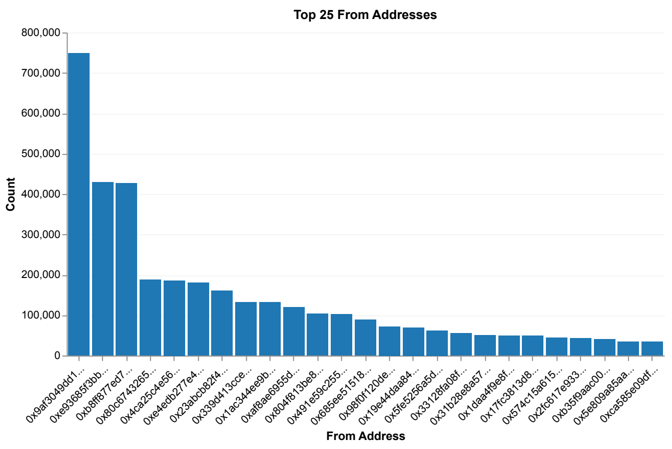

Figure 8: Top 25 from addresses

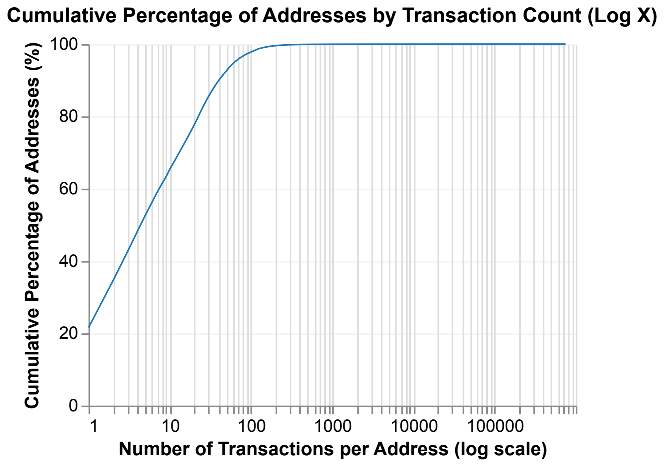

Figure 9: Cumulative distribution of number of transactions per from address

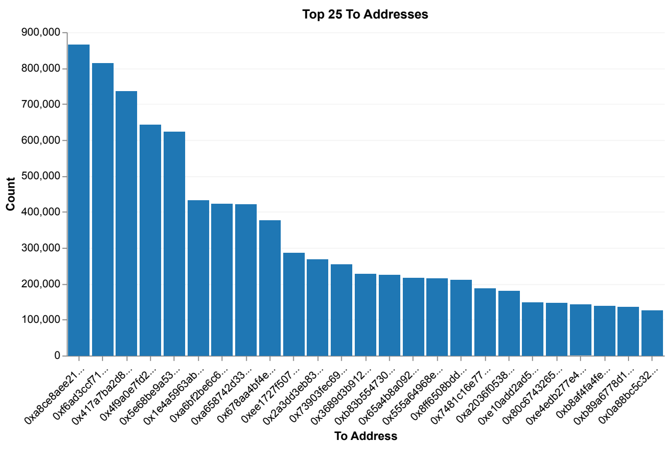

Figure 10: Top 25 to addresses

Finally, Figure 12 presents the top 25 most called functions. We observe that the top use on this network corresponds to a native transfer (`0x` as the input of the transaction), followed by the calls for these function headers (obtained from

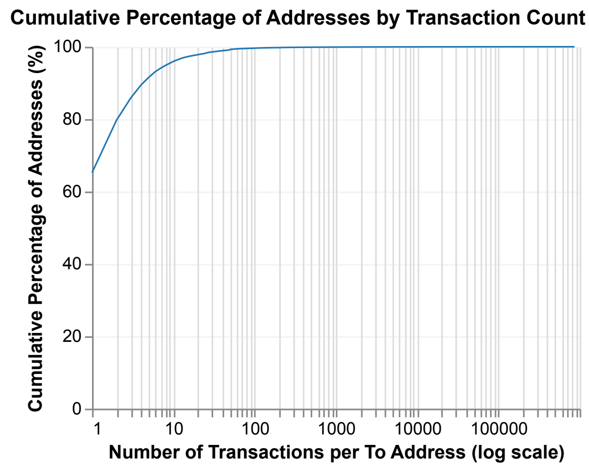

Figure 11: Cumulative distribution of number of transactions per to address (zoomed)

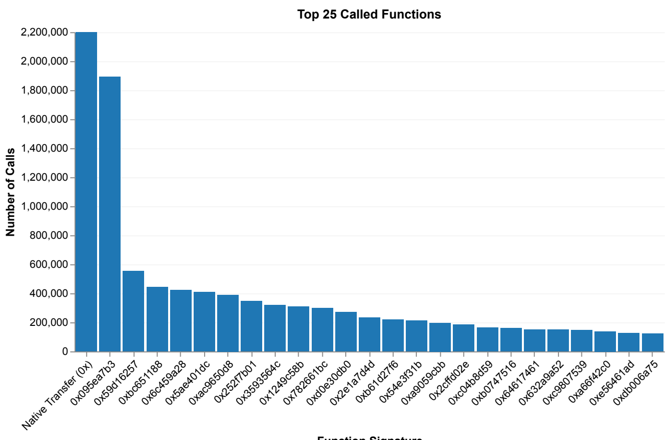

Figure 12: Top 25 function calls

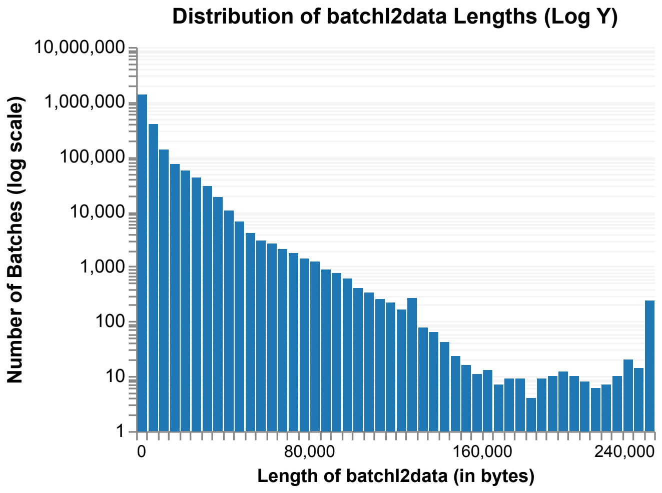

Figure 13: Histogram of batchl2data by lenght of it.

4byte.directory[3]):

`0x095ea7b3` : `approve()`, is the standard ERC-20 token approval function. It allows a token holder to grant permission to another address (spender) to transfer a specified amount of tokens on their behalf.

`0x59d16257` : `claim()`, is a rewards claiming function commonly used in DeFi protocols, airdrops, and staking systems.

`0x6c459a28` : `execute()`, is a generic execution function often found in multisig wallets, governance contracts, or proxy contracts.

`0x5ae401dc` : `multicall()`, is Uniswap V3's multicall function that enables batching multiple function calls into a single transaction.

4.3. batchL2data metrics

In this Section, we analyze and discuss the charts related to `batchL2data`, also known as *Data Availability*. This field represents the data of all the transactions contained in a batch, that is sent to the Layer 1 in order to provide with a permissionless way to access the state updates on the rollup.

Figures 13 and 14 represent an histogram of `batchL2data` (on a logarithmic scale) in terms of bytes, the daily mean, and daily sum. The histogram shows a long-tailed distribution closely following a power-law behavior (fitted power law alpha 3.777 with a Kolmogorov-Smirnov distance of 0.0151, see Figure 24). This indicates that, in general, the sequencer and prover of this network are bundling the vast majority of batches with a low amount of Data Availability, rather than sending large amounts of bytes to the L1.

Regarding the Figure representing the daily mean and sum (Figure 14), we observe that the sum is rather close to both the daily transaction graph (Figure 20), as well as the daily sum of size of the transaction. Additionally, the mean Figure presents a graph that can be divided into 4 zones: from genesis to early June 2023, from June 2023 to mid February 2024, mid February 2024 to October 2024, and from October 2024 to the last day of 2024. To the best of our knowledge, those changes on the tendencies in this Figure are closely related to changes on the prover of the network.

4.4. Analyzing the bundling of transactions into blocks and batches

In this Section, we present the graphs and analysis of the data related to how the rollup is performing the bundling. As we stated in Section 3.1, this rollup bundles L2 transactions into L2 blocks, and those blocks are bundled again into L2 batches. Those batches are used to update the state root and are the input

[3]Function calls in the Ethereum Virtual Machine are specified by the first four bytes of data sent with a transaction. These 4 byte signatures are defined as the first four bytes of the Keccak hash (SHA3) of the canonical representation of the function signature. 4byte.directory hosts a database that maps the 4 byte representation of the function header, with a human-readable representation of it.

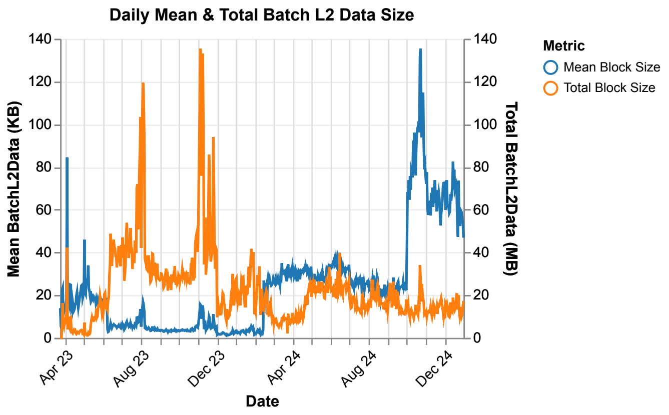

Figure 14: Mean and sum of batchl2data per day

for the validity proof of correct state transition. Let us explore how this rollup is bundling transactions into blocks and batches.

Figure 15 represents the mean count of transactions per block, grouped by day. Consistent with earlier observations, we see that from the genesis block up until mid February 2024, this rollup operated such that each block contained exactly one transaction. After this period, the network exhibits a more typical pattern for a general-purpose blockchain, with the mean transaction count varying over time. Notably, the mean remains below one, indicating the presence of numerous empty blocks.

Figure 16 presents a histogram that groups the batches by the number of transactions they hold. Interestingly enough, in this histogram, we can see that the vast majority of batches were bundling between 0 and 5 transactions. In fact, if you consider batches that contain 0 and 1 transactions, we cover more that 95 % of the total number of batches in this study.

Similarly, Figure 17 presents a histogram (on logarithmic scale) that shows the number of blocks per batch. Again, we can see a long tail function following a power-law pattern (fitted power law alpha of 2.201, with a Kolmogorov-Smirnov distance of 0.0271, see Figure 23), indicating that the executor on this rollup is more comfortable batching few blocks per batch, rather than bundling together a huge amount of blocks and transactions per batch.

Finally, in this Section we compare the bundling of transactions in blocks against the bundling of blocks in batches. Figure 18 presents a chart where each point corresponds to a batch, the X-axis counts the number of transactions a particular batch has, while the Y-axis counts the number of blocks a particular batch has. As expected, we can clearly see a strong concentration on the $y = x$ line (due to the period of time where 1 block was equal to 1 transaction). In the Figure, this corresponds to both Initial Period (in blue), and Low Activity (in green). This line is particularly interesting because it divides the region in two parts: the region above this line corresponds to those batches that bundle more blocks into batches than transactions into blocks, while the region below this line corresponds to those batches that bundle more transactions into blocks than blocks into batches. We can clearly see that this rollup prefers to bundle more blocks into batches, than to transactions into blocks, indicating that the recursive proof (i.e. the proof that aggregate blocks together into batches) of the system was on a more mature state when compared to the leaf proof (i.e. the proof that aggregate transactions together into blocks).

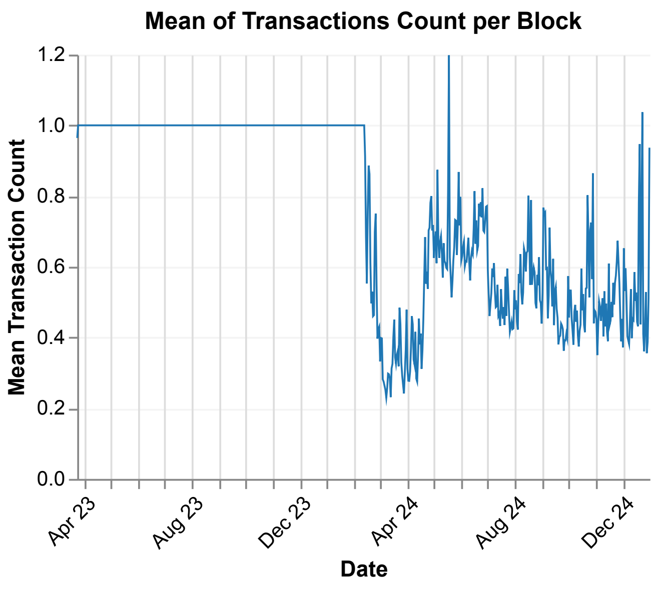

Figure 15: Mean Transaction Count per Block, grouped by day

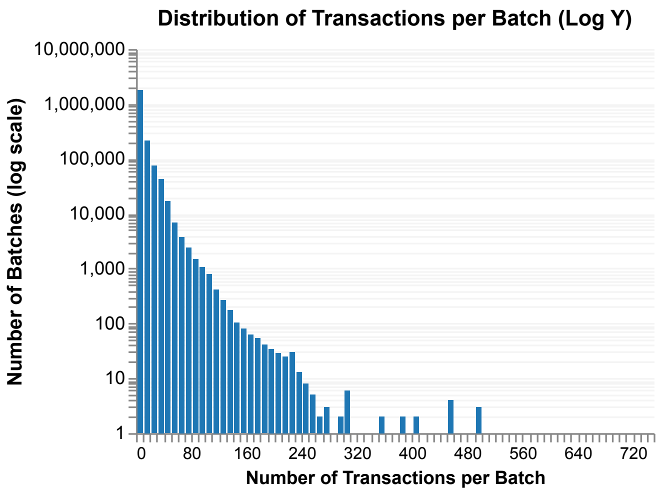

Figure 16: Histogram in log scale of transaction count per batch

4.5. *Sequencer Metrics*

Figure 19 shows a histogram of the Sequencers that produced each block on the network (the field `miner` on the block). Polygon stated clearly that this network is run over a single –centralized– Sequencer (address `0x148ee7daf16574cd020afa34cc658f8f3fbd2800`), but we found that the genesis block was mined by the address `0x0000000000000000000000000000000000000000` and we found that 273 blocks were mined by the address `0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266`. All of these blocks were mined on 31 August 2023, between 08:54AM and 09:04AM, the same day the Polygon zkEVM

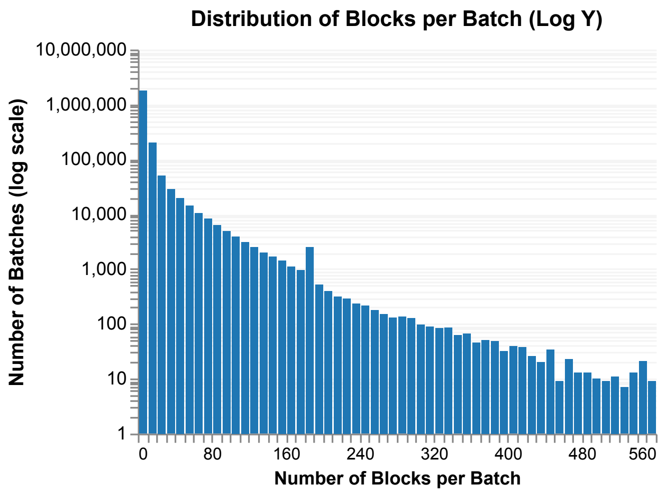

Figure 17: Histogram in log scale of block count per batch

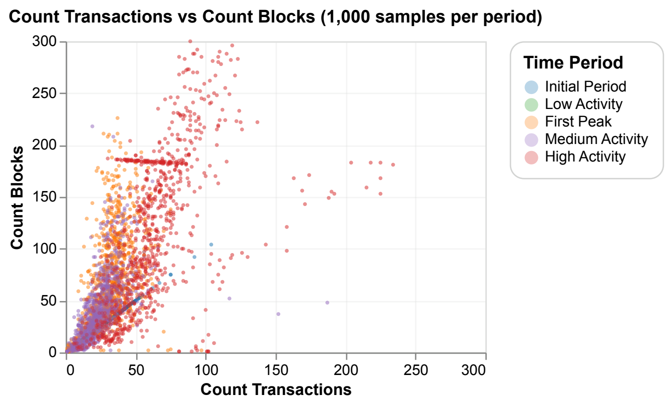

Figure 18: Blocks vs. Batches Plot

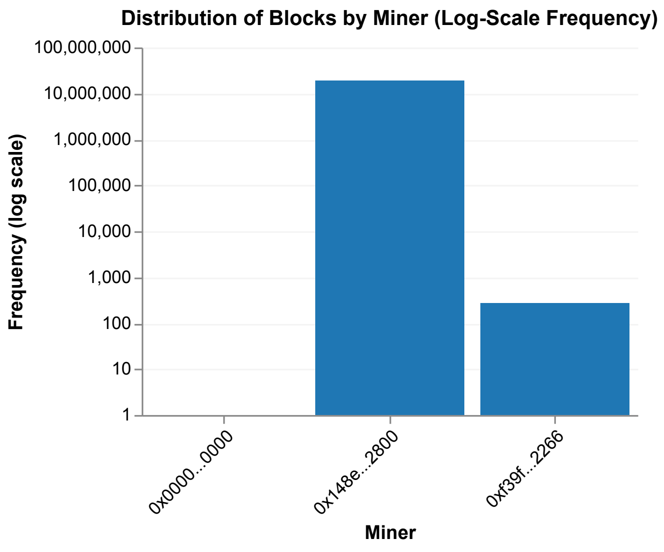

Figure 19: Distribution of blocks by producer

team deployed an update on both the node and the prover [39]. This change in the sequencer was, most likely, due to issues when deploying that particular update.

4.6. Miscellaneous Metrics

For the sake of completeness, we have explored and considered all the fields of the data collected. We did not find any relevant information associated with the following fields, which are retained for reasons of backward compatibility. Specifically, for blocks, the fields `difficulty`, `totaldifficulty`, `extradata`, and `logsbloom` contain no meaningful information; and for receipts, the fields `logs`, `logsbloom` contain no meaningful information.

5. Conclusions and Future Work

Reliable access to on-chain records remains a major bottleneck for empirical work. Turning raw blockchain traces into analysis-ready data typically requires infrastructure, careful decoding, and significant preprocessing challenges that are magnified in Layer 2 systems such as Polygon zkEVM.

Our response to this challenge is a curated Polygon zkEVM dataset derived from an archive node and normalized for analysis. It includes coherent tables for blocks, transactions, receipts, and batches, as well as qualitative analysis of the extracted data.

We make our code, notebooks, and dataset publicly available on GitHub `Cite redacted for submission.` and CORA [30]. We invite contributions, issue reports, and extensions. Our intention is to reduce the entry cost of Layer 2 research and to help the community build in a shared, well-documented foundation.

The public availability of this dataset opens up several research directions. We highlight some key opportunities below. We believe that it would be valuable to quantify how scaling solutions contribute to system scalability across multiple dimensions, including computational efficiency, energy consumption, and storage requirements. Such an analysis could provide insights on the overall impact of these technologies on network performance and resource utilization.

Additional future work directions include conducting quantitative evaluations comparing different scaling approaches and their respective trade-offs, as well as investigating user behavior patterns and adoption dynamics within these systems. These research avenues could contribute to a more comprehensive understanding of scaling solution effectiveness and their role in blockchain ecosystem evolution.

6. Acknowledgments

References

[1] Layer 2. URL https://ethereum.org/en/layer-2/.

[2] Ethereum Archive Node. URL https://ethereum.org/en/developers/docs/nodes-and-clients/archive-nodes/.

[3] Understanding the Role of Archival and Pruned Nodes in the Decentralization of Bitcoin - D-Central, March 2022. URL https://d-central.tech/understanding-the-role-of-archival-and-pruned-nodes-in-the-decentralization-of-bitcoin/. Section: Bitcoin.

[4] Hardware requirements. URL https://geth.ethereum.org/docs/getting-started/hardware-requirements.

[5] BRUCE JAY NELSON. *Remote Procedure Call.* Ph.D., 1981. URL https://www.proquest.com/docview/303093578/abstract/1DF38861056D43C7PQ/1. ISBN: 9798617016606.

[6] etherscan.io. Ethereum (ETH) Blockchain Explorer. URL https://etherscan.io/.

[7] arbiscan.io. Arbitrum One (ETH) Blockchain Explorer. URL http://arbiscan.io/.

[8] Polygon zkEVM | Scaling for the Ethereum Virtual Machine, . URL https://polygon.technology/polygon-zkevm.

[9] Inconsistencies in PolygonScan zkEVM Published Data | MigaLabs - Blockchain Ecosystem Observatory, . URL https://migalabs.io/blog/inconsistencies-in-polygon-scan-zk-evm-published-data.

[10] Maximal extractable value (MEV). URL https://ethereum.org/en/developers/docs/mev/.

[11] Christos A. Makridis, Michael Fröwis, Kiran Sridhar, and Rainer Böhme. The rise of decentralized cryptocurrency exchanges: Evaluating the role of airdrops and governance tokens. *Journal of Corporate Finance*, 79:102358, April 2023. ISSN 0929-1199. doi: 10.1016/j.jcorpfin.2023.102358. URL https://www.sciencedirect.com/science/article/pii/S092911992300007X.

[12] Yongge Wang. Automated Market Makers for Decentralized Finance (DeFi), May 2024. URL http://arxiv.org/abs/2009.01676. arXiv:2009.01676.

[13] Krzysztof Gogol, Robin Fritsch, Malte Schlosser, Johnnatan Messias, Benjamin Kraner, and Claudio Tessone. Liquid Staking Tokens in Automated Market Makers, July 2024. URL http://arxiv.org/abs/2403.10226. arXiv:2403.10226.

[14] Taishi Nakai, Akira Sakurai, Shiori Hironaka, and Kazuyuki Shudo. The Blockchain Trilemma Described by a Formula. In *2023 IEEE International Conference on Blockchain (Blockchain)*, pages 41–46, December 2023. doi: 10.1109/Blockchain60715.2023.00016. URL https://ieeexplore.ieee.org/abstract/document/10411444. ISSN: 2834-9946.

[15] A rollup-centric ethereum roadmap, October 2020. URL https://ethereum-magicians.org/t/a-rollup-centric-ethereum-roadmap/4698.

[16] Shafi Goldwasser, Silvio Micali, Chales Rackoff, and University of Toronto. The knowledge complexity of interactive proof-systems. In Weizmann Institute of Science and Oded Goldreich, editors, *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. Association for Computing Machinery, October 2019. ISBN 978-1-4503-7266-4. doi: 10.1145/3335741.3335750. URL https://dl.acm.org/citation.cfm?id=3335750.

[17] The different types of ZK-EVMs. URL https://vitalik.eth.limo/general/2022/08/04/zkevm.html.

[18] L2BEAT - The state of the layer two ecosystem. URL https://l2beat.com/scaling/summary.

[19] Jens Groth. On the Size of Pairing-Based Non-interactive Arguments. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016*, volume 9666, pages 305–326. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016. ISBN 978-3-662-49895-8 978-3-662-49896-5. doi: 10.1007/978-3-662-49896-5_11. URL http://link.springer.com/10.1007/978-3-662-49896-5_11. Series Title: Lecture Notes in Computer Science.

[20] Xiaoqiang Sun, F. Richard Yu, Peng Zhang, Zhiwei Sun, Weixin Xie, and Xiang Peng. A Survey on Zero-Knowledge Proof in Blockchain. *IEEE Network*, 35(4): 198–205, July 2021. ISSN 1558-156X. doi: 10.1109/MNET.011.2000473. URL https://ieeexplore.ieee.org/document/9520375/.

[21] Maria Inês Silva, Johnnatan Messias, and Benjamin Livshits. A Public Dataset For the ZKsync Rollup, February 2025. URL http://arxiv.org/abs/2407.18699. arXiv:2407.18699 [cs].

[22] Kamil Kaczyński and Aleksander Wiącek. Anomaly Detection in ZkSync Transactions with Unsupervised Machine Learning. pages 565–570, October 2025. ISBN 978-989-758-760-3. URL https://www.scitepress.org/Link.aspx?doi=10.5220/0013441600003979.

[23] Christof Ferreira Torres, Albin Mamuti, Ben Weintraub, Cristina Nita-Rotaru, and Shweta Shinde. Rolling in the Shadows: Analyzing the Extraction of MEV Across Layer-2 Rollups. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, pages 2591–2605, New York, NY,

USA, December 2024. Association for Computing Machinery. ISBN 979-8-4007-0636-3. doi: 10.1145/3658644.3690259. URL https://dl.acm.org/doi/10.1145/3658644.3690259.

[24] Rainer Feichtinger, Florian Grötschla, Lioba Heimbach, and Roger Wattenhofer. Benchmarking GNNs Using Lightning Network Data, July 2024. URL http://arxiv.org/abs/2407.07916. arXiv:2407.07916 [cs].

[25] Bitcoin Lightning Network Stats Dataset | Spreadsheet Download | Gigasheet. URL https://www.gigasheet.com/sample-data/bitcoin-lightning-network-stats-dataset.

[26] 0xPolygon/cdk-erigon: Ethereum implementation on the efficiency frontier. URL https://github.com/0xPolygon/cdk-erigon.

[27] rpc_utils.py. URL https://github.com/0xAdriaTorralba/PolygonZKEVMAnalysis/blob/main/src/utils/rpc_utils.py.

[28] Polars, . URL https://www.pola.rs/.

[29] Vega-Altair: Declarative Visualization in Python — Vega-Altair 5.5.0 documentation. URL https://altair-viz.github.io/.

[30] Repositori de dades de recerca. URL https://dataverse.csuc.cat/previewurl.xhtml?token=be7480ff-690d-405e-82da-c2fb9f40613e.

[31] Submit transactions - Polygon Knowledge Layer. URL https://docs.polygon.technology/zkEVM/architecture/protocol/transaction-life-cycle/submit-transaction/#transactions-and-blocks-on-zkevm.

[32] JSON-RPC - Polygon Knowledge Layer. URL https://docs.polygon.technology/zkEVM/architecture/proving-system/json-rpc-to-proof/#etrog-upgrade-forkid-6.

[33] Polygon [@0xPolygon]. A historic moment as @VitalikButerin made his first transaction on Polygon #zkEVM Mainnet Beta http://go.polygon.technology/zkevm-launch https://t.co/Ua6kKknudW, March 2023. URL https://x.com/0xPolygon/status/1640461759785103360.

[34] Dragonfruit Fork. URL https://docs.polygon.technology/zkEVM/architecture/proving-system/json-rpc-to-proof/#dragonfruit-upgrade-forkid-5.

[35] Inca Berry Upgrade, . URL https://polygon.technology/blog/polygon-zkevm-inca-berry-upgrade-coming-to-mainnet-beta.

[36] EIP-4844: Proto-Danksharding. URL https://www.eip4844.com/.

[37] Eggfruit Upgrade Live: The cdk-erigon Sequencer is Live on Polygon zkEVM Mainnet Beta. URL https://polygon.technology/blog/eggfruit-upgrade-incoming-polygon-zkevm-mainnet-beta-will-see-the-cdk-erigon-sequencer-go-live.

[38] 0xPolygon/cdk-erigon, September 2025. URL https://github.com/0xPolygon/cdk-erigon. original-date: 2023-11-29T11:22:05Z.

[39] Historical data - Polygon Knowledge Layer. URL https://docs.polygon.technology/zkEVM/get-started/historical-data/#31st-aug-2023.

7. Appendix

In this Section, we include additional figures that do not fit on the manuscript, but can be helpful when analyzing the dataset.

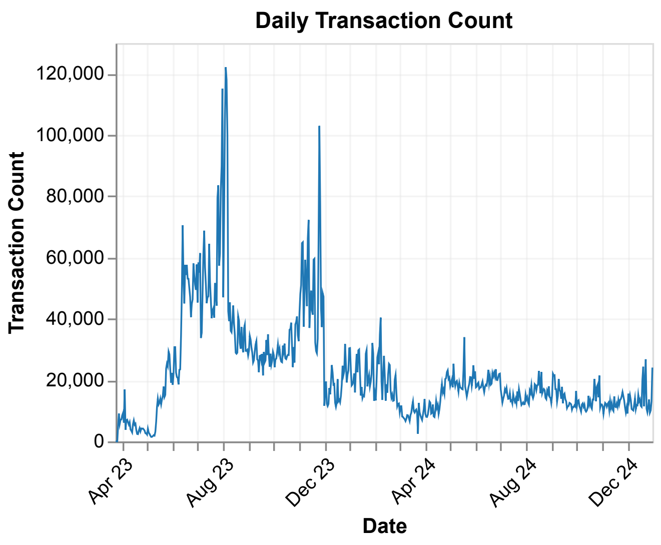

Figure 20: Daily Transactions Count

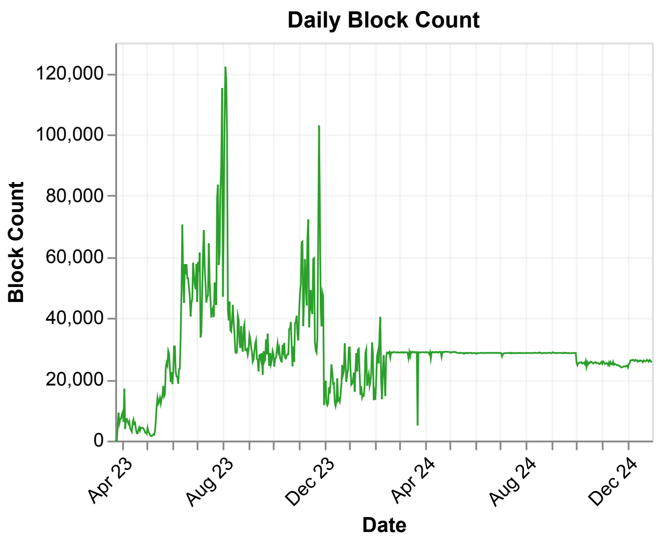

Figure 21: Daily Blocks Count

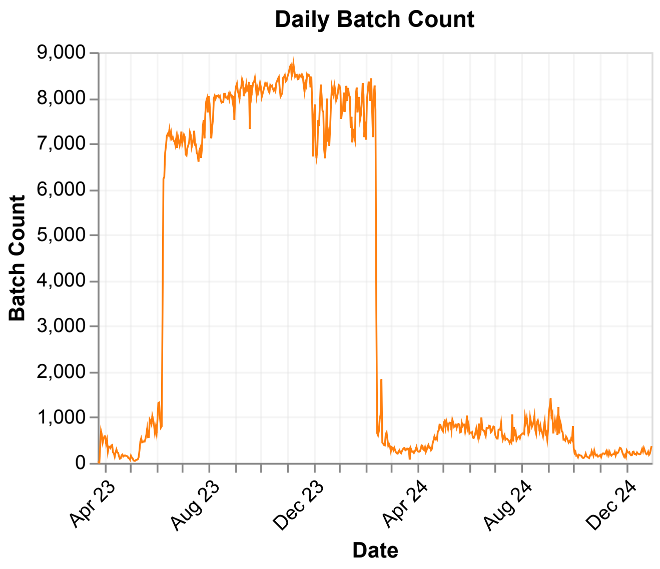

Figure 22: Daily Batches Count

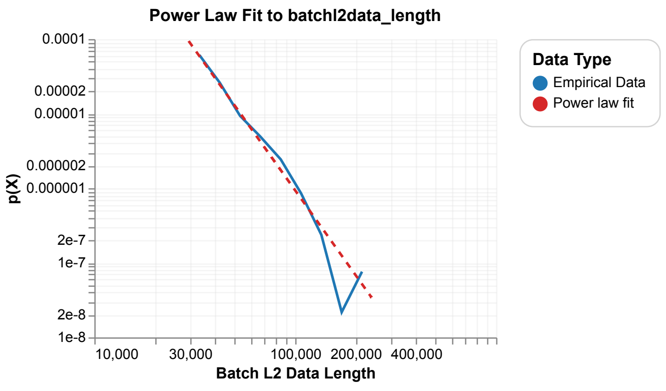

Figure 24: Powerlaw fit to batchl2data.

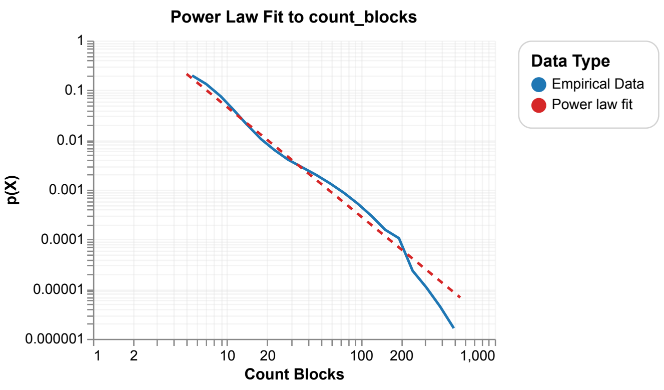

Figure 23: Powerlaw fit to blocks per batch.