



AGH UNIVERSITY OF KRAKOW
FACULTY OF METALS ENGINEERING AND INDUSTRIAL COMPUTER SCIENCE
DEPARTMENT OF APPLIED COMPUTER SCIENCE AND MODELLING

Diploma project

*Exploring Ethereum network scalability: an examination of
Layer 2 implementations*

*Badanie skalowalności sieci Ethereum – analiza rozwiązań opartych na
warstwie 2*

Author:	Hubert Józef Kotliński
Field of study:	Computational Engineering
Thesis supervisor:	Krzysztof Bzowski, PhD Eng.

Kraków, 2024

Contents

1.	Introduction.....	4
2.	Overview of issues.....	8
2.1	Scalability problem explanation – the blockchain trilemma in the Ethereum case	8
2.2	Overview of potential scalability options	9
2.3	Optimistic rollup vs ZK rollup.....	12
3.	Technical aspects of the analysis of selected solutions	16
3.1	Risk analysis	16
3.2	Decentralization stage.....	16
3.3	Upgradeability	17
3.4	Arbitrum One analysis	18
3.4.1	Introduction to Arbitrum One	18
3.4.2	Arbitrum One risk analysis	19
3.4.3	Arbitrum One decentralization assessment.....	20
3.4.4	Arbitrum One upgrade pathways	21
3.5	Base analysis.....	22
3.5.1	Introduction to Base.....	22
3.5.2	Base risk analysis.....	24
3.5.3	Base decentralization assessment	25
3.5.4	Base upgrade pathways.....	26
3.6	zkSync Era analysis	27
3.6.1	Introduction to zkSync Era	27
3.6.2	zkSync Era risk analysis	28
3.6.3	zkSync Era decentralization assessment.....	29
3.6.4	zkSync Era upgrade pathways	30
4.	A brief introduction to additional resources to consider.....	32
4.1	Worldcoin – building on L2 instead of L1	32

4.2	EIP-4844	33
5.	Design of tools for data collection	35
6.	Implementation details.....	37
6.1	Address generation	37
6.2	Sending multiple transactions between two addresses	37
6.3	Swapping tokens on Uniswap.....	39
6.4	Transitioning from L1 to L2 via native bridge	41
6.5	Returning from L2 to L1 via native bridge.....	42
6.6	Retrieving data from the L2BEAT API.....	43
6.7	Retrieving data from the Dune Analytics	44
7.	Results.....	46
7.1	Economic analysis of rollups	46
7.2	Analysis of activity on chosen networks	49
7.3	Analysis of user experience on chosen networks	51
8.	Summary and conclusions	54

1. Introduction

In recent years, the blockchain industry has witnessed a profound transformation. Once popular networks have largely faded, with their native tokens losing much of their value. Ethereum¹ stands as an exception to this trend, having achieved success through a strategic approach to the tokenomics of its native token, Ether². More importantly, it has taken a novel approach to rectify its most pressing issue – scalability amidst a critical mass of users. Unlike competing chains like Solana³ that focus on code modifications at the base layer, or L1 (Layer 1), Ethereum has fostered an open market for solutions that involve creating numerous "child" chains. On these chains, users can carry out the majority of their transactions with relatively low costs and faster execution, while theoretically still enjoying the security properties of Ethereum itself. Such solutions are termed L2 (Layer 2). The absence of an official L2 solution from Ethereum's core team has initiated a sort of experimental arms race among developer teams from every corner of the globe.

A pivotal moment in this evolutionary journey was the announcement (Figure 1.) on October 2, 2020, of the "rollup-centric Ethereum roadmap" [1] by Vitalik Buterin⁴. This proclamation shifted the Ethereum team's focus towards implementing enhancements in the main network's protocol, tailor-made for L2 solutions. This initiative has continued to the present day, with the adoption of these solutions being recognized as the answer to the network's scalability issues in the short to medium term.

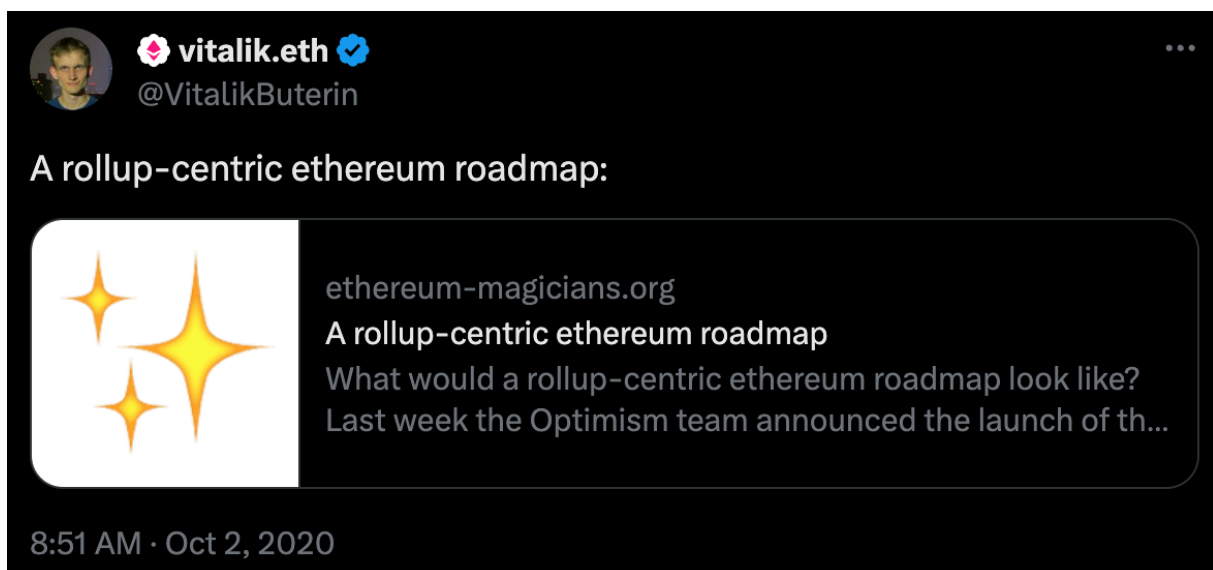


Figure 1. Vitalik Buterin announces rollup-centric Ethereum roadmap

Source: x.com (January 4, 2024)

Since that declaration, the amount of funds secured on L2 networks has surged from approximately 7 million dollars to over 22 billion dollars (Figure 2.). Moreover, there are over

¹ <https://ethereum.org/>

² <https://www.investopedia.com/terms/e/ether-cryptocurrency.asp>

³ <https://solana.com/>

⁴ <https://cointelegraph.com/learn/who-is-vitalik-buterin>

30 projects with an active public mainnet, and dozens more in the testing phase, vividly demonstrating the scale of interest that the industry is witnessing.

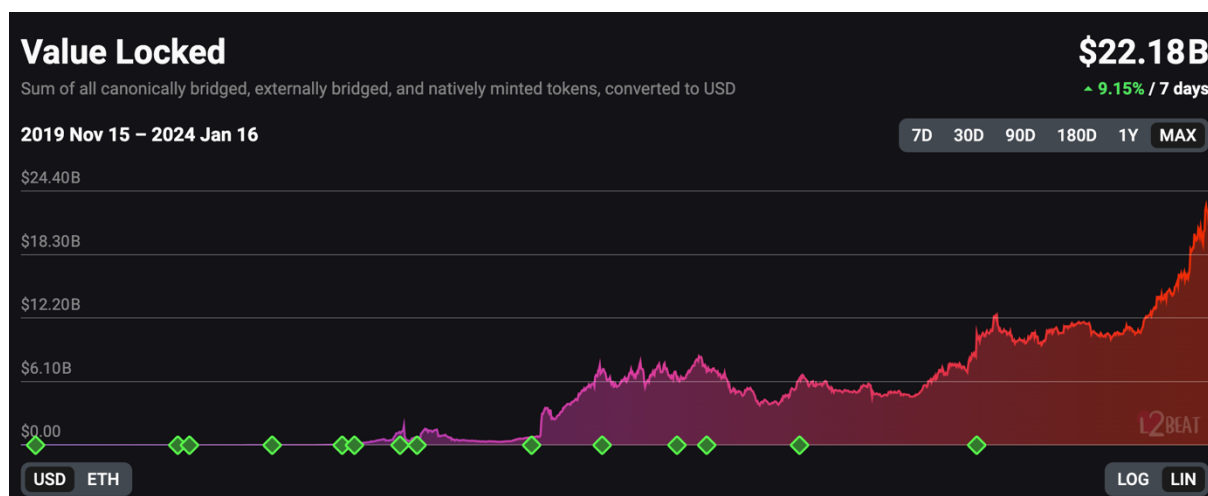


Figure 2. Chart of total value in USD of all tokens on L2 networks
Source: l2beat.com (January 16, 2024)

It is also important to note that the TVL⁵ (Total Value Locked) of L2 solutions already surpassed the TVL of all other L1s (Figure 3.), which makes them even more interesting topics to research.

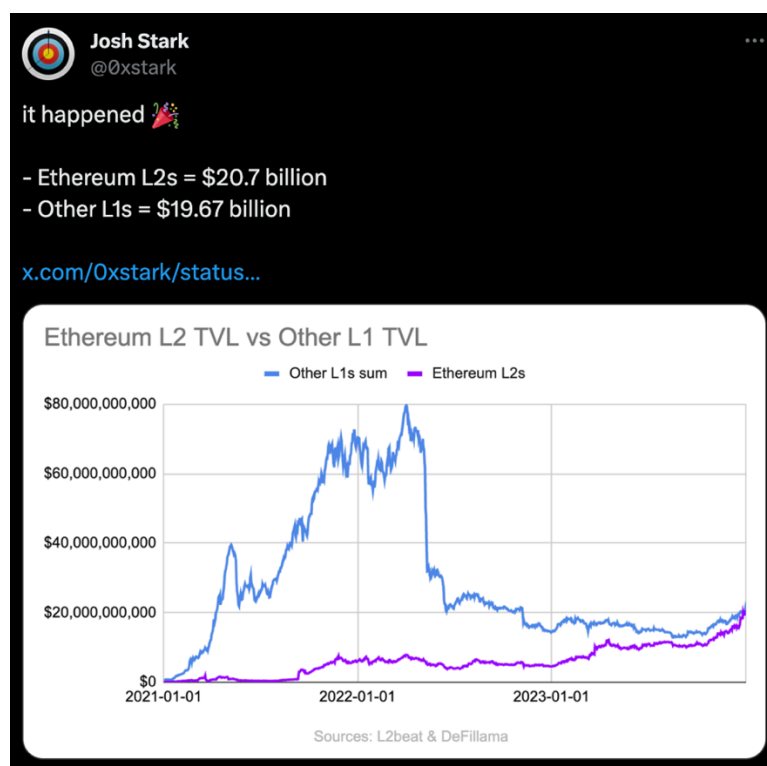


Figure 3. Josh Stark commenting L2 vs L1 TVL
Source: x.com (January 4, 2024)

⁵ <https://medium.com/l2beat/redefining-total-value-locked-for-l2s-756160602747>

The diversity of approaches to the structure of such networks has proven to be so extensive that even experienced users struggle to analyze each of the solutions available on the market. This presents an additional problem as they are encouraged to utilize several such networks simultaneously to optimize their experience. The differences between these networks are not limited to their TVL but also include their technological outline, degree of decentralization, potential risks of fund loss, methods of implementing changes in their main protocol, and the maximum number of transactions they can handle. Some are also created with specific use cases in mind, such as NFTs (Non-Fungible Tokens), gaming, or DeFi (Decentralized Finance). Numerous debates arise over which projects can truly be considered legitimate L2 solutions and which cannot.

The author of the paper defines Layer 2 as any network, system, or technology outside of the blockchain (off-chain) that is built on top of a blockchain (also known as a Layer 1 network), helping to enhance the capabilities of the underlying base network **while inheriting its security**. This means that transaction data must be verified and confirmed by the underlying blockchain network, not by a separate set of nodes. For example, the author does not consider sidechains to be Layer 2 because they typically implement their own consensus mechanisms and validators⁶, leading to a different set of security guarantees than those offered by the base chain.

The objective of this thesis is to analyze several selected Layer 2 projects and assess their potential to effectively scale the Ethereum network over the next few years. The author focuses on collecting an appropriate set of data and technical details, utilizing various sources, which will enable a thorough evaluation of these projects in terms of their real-world future application, adoption, relevance to the market in generating revenue, and determining whether the rollup-centric Ethereum roadmap has been a success or not and if it has truly enabled Ethereum to overcome its most significant challenge.

Below are the detailed elements that comprise the scope of the thesis:

1. **Selection of projects for analysis.** For this thesis, in addition to the main Ethereum network, the following projects have been selected:
 - Arbitrum One⁷
 - Base⁸
 - zkSync Era⁹

This decision was made due to their market dominance and long-standing presence (for a very young industry), which ensures easy access to detailed documentation and a wealth of data from their testnets and mainnets via various external applications used. They also represent the most popular and, in the private opinion of the author of this thesis, the safest Layer 2 technologies known as rollups - optimistic rollups and ZK rollups. While the presented techniques can be applied to other projects, there might be issues with accessing some data. In this work, uniformity of approach is the only way to ensure that each project is evaluated fairly and impartially. Moreover, a significant portion of other projects is built on the codebase of these technologies, which facilitates

⁶ <https://coinmarketcap.com/academy/glossary/validator>

⁷ <https://arbitrum.io/>

⁸ <https://base.org/>

⁹ <https://zksync.io/>

convenient and extensive future development of implementations with additional examples.

2. **Data sources selection.** The choice of appropriate data sources will include both a custom script for executing transactions and recording their data simulating single user experience, as well as data retrieval using L2BEAT¹⁰ and Dune Analytics¹¹ due to their reputation and the rich selection of data for the projects in focus.
3. **Presentation of Layer 2 technology theories and analysis of implementations from both a technical perspective and that of a project targeting mass adoption.** This includes describing the operational outline of Layer 2 technologies, with particular emphasis on optimistic rollup and ZK rollup technologies, their associated risks, levels of decentralization, and upgradeability. Additionally, Worldcoin¹² is cited as an exemplary project that aligns with the concept of building on Layer 2.
4. **Design and implementation of tools to simulate the experience of a standard user and for data extraction from websites.** A tool has been created for the automatic execution of transactions, aiming to simulate the behavior of a random individual user who wishes to engage with the services of selected Layer 1 and Layer 2 projects during periods of typical network traffic. Additionally, scripts have been developed for retrieving essential data on overall network activity from specified websites. The simulation tool was developed using Node.js¹³ and ethers.js¹⁴. For Dune Analytics, SQL (Structured Query Language) queries and their charting tools were employed, while the L2BEAT data was accessed using their API (Application Programming Interface) in conjunction with Python.
5. **Data collection and processing.** Gathering data from one's tool and scripts and preparing them for in-depth analysis. This process involves performing the appropriate queries, downloading data, and its visualization.
6. **Analysis of the collected data and drawing conclusions.** A summary of the research findings. The collected data will be used to answer the following questions:
 - Are L2 projects capable of realistically scaling the Ethereum network and enabling its broader adoption?
 - Has the rollup-centric Ethereum roadmap proven to be a correct decision?
 - Are L2 projects a long-term solution to scalability issues, or merely a temporary fix?

Additionally, the thesis includes a discussion on further research in this area and the possible future of this technology.

This scope ensures a comprehensive approach to the subject matter, from the technical aspects of data collection and processing to the theoretical and practical analysis of the results obtained.

¹⁰ <https://l2beat.com/scaling/summary>

¹¹ <https://dune.com/browse/dashboards>

¹² <https://worldcoin.org/>

¹³ <https://nodejs.org/en>

¹⁴ <https://docs.ethers.org/v5/>

2. Overview of issues

2.1 Scalability problem explanation – the blockchain trilemma in the Ethereum case

Ethereum stands as a pioneering public blockchain, notably as the first to implement what is recognized as smart contracts [2]. These smart contracts are essentially programs that operate on the Ethereum blockchain. Conceptually, this is akin to running a program on a personal computer; however, instead of a single computer, these programs execute on a distributed network of computers, with validator nodes managing the operations. For the execution of these programs, transactions are sent to a specific address accompanied by the necessary data. Each Ethereum block has a restricted capacity for recording the output of these programs. To deter attacks that fill blocks with meaningless transactions, a gas fee system has been established, charging for each transaction in the native token, Ether. This fee fluctuates with demand - higher demand leads to higher fees [3].

Unfortunately, due to the ever-increasing interest in the Ethereum network and the influx of new smart contracts - desired by a growing user base - the network has reached its maximum capacity of approximately 12 transactions per second (Figure 4.). At times of peak interest, such as during the recent expansion of the restaking limit on Eigen Layer¹⁵, gas fees can spike to exorbitant rates, sometimes exceeding 0.05 ETH (about 125 USD at rates from January 16, 2024) per transaction, regardless of the transaction's value.

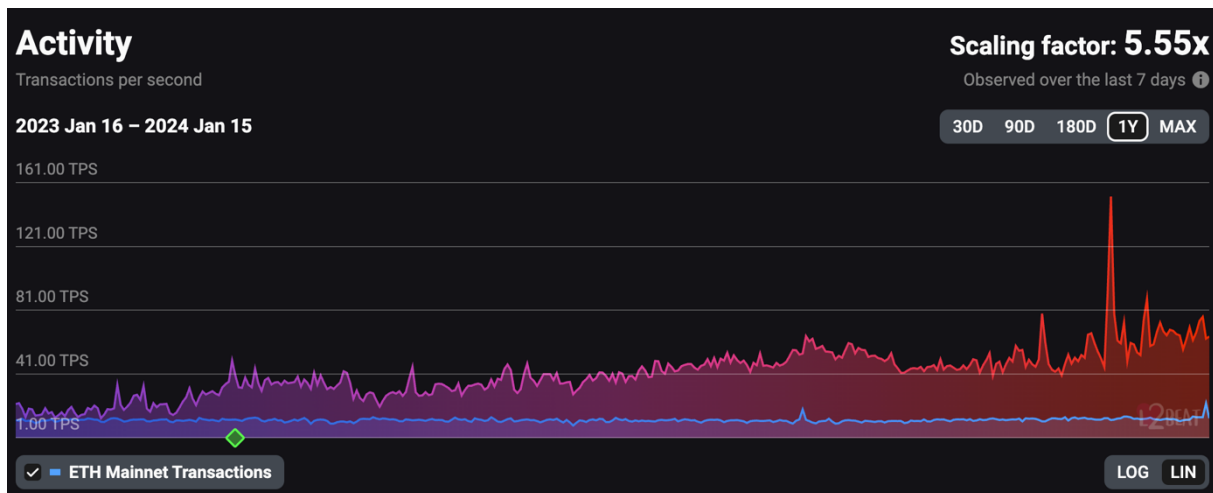


Figure 4. Chart of TPS (Transactions Per Second) on Ethereum mainnet and all L2 networks for last year
Source: l2beat.com (January 16, 2024)

Ethereum, therefore, finds itself grappling with the blockchain trilemma (Figure 5.) like every other decentralized network. This trilemma suggests that a public blockchain system cannot simultaneously achieve the optimal levels of decentralization, security, and scalability. Scalability refers to a blockchain's capacity to handle a large volume of transactions, security pertains to its resilience against various forms of cyber-attacks and its defense against double-spending, and decentralization ensures that control over the network is not held by a few

¹⁵ <https://www.eigenlayer.xyz/>

entities. The intrinsic design of decentralized networks often implies that attempts to enhance scalability can diminish decentralization or security. Given that decentralization and security are among Ethereum's most valued qualities, the community has been actively seeking methods to improve scalability without compromising these critical aspects [4].

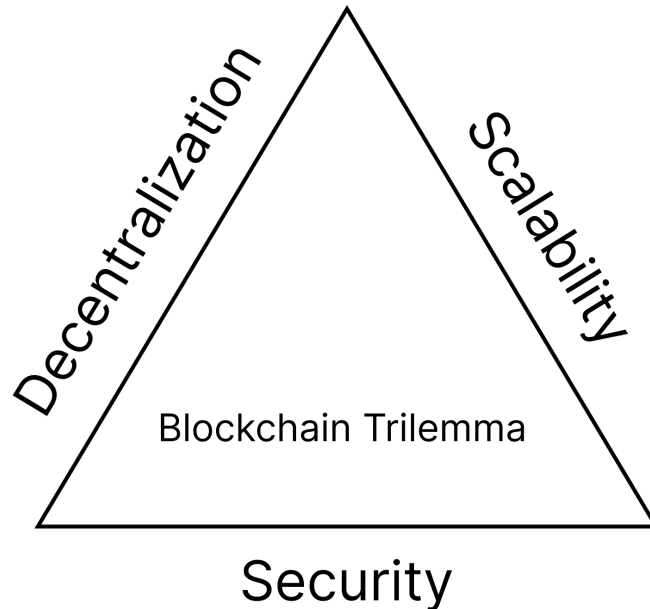


Figure 5. The blockchain trilemma visualization
Source: Author's own work

2.2 Overview of potential scalability options

According to the perspectives shared by Vitalik Buterin, there exist two primary strategies for expanding the capacity of a blockchain ecosystem [5]:

1. The first strategy involves enhancing the blockchain's inherent capacity to process a greater number of transactions, a method referred to as on-chain scaling. However, as earlier outlined, such rapid modifications could undermine the network's other two critical qualities. Despite this, they are feasible and are continuously being explored and developed by the Ethereum core team. Nevertheless, the actualization of these strategies is anticipated to require several years, during which Ethereum's progress could stall, potentially allowing rival platforms to gain the upper hand.
2. The second strategy suggests altering how the blockchain is utilized by its users. Rather than conducting all operations directly on the blockchain, users could carry out most of their activities off-chain, that is, outside the blockchain's direct system, only returning to Ethereum for tasks that necessitate its unique advantages. Depending on the approach taken, a variety of solutions could be distinguished. One of the groups of those solutions is referred to as Layer 2.

Layer 2 technology is still in its nascent stages, and as such, there is ongoing discourse regarding the precise definition of a Layer 2 solution and which methodologies best serve the

network and its users. The author of this document intends to offer a personal viewpoint on this topic. It is crucial to acknowledge that given the experimental and swiftly evolving nature of this technology, the assertions made in this document may lose their relevance after some time has passed since its release.

It is crucial to clarify that any so-called solution that fails to publish data on L1, in our case Ethereum, and consequently introduces supplementary trust assumptions does not qualify as an L2 solution. Nevertheless, such solutions have been in existence since the inception of scalability research and should not be disregarded. This is particularly true in light of recent advancements in data availability technologies (Figure 6.) like Celestia¹⁶ and Eigen DA¹⁷.

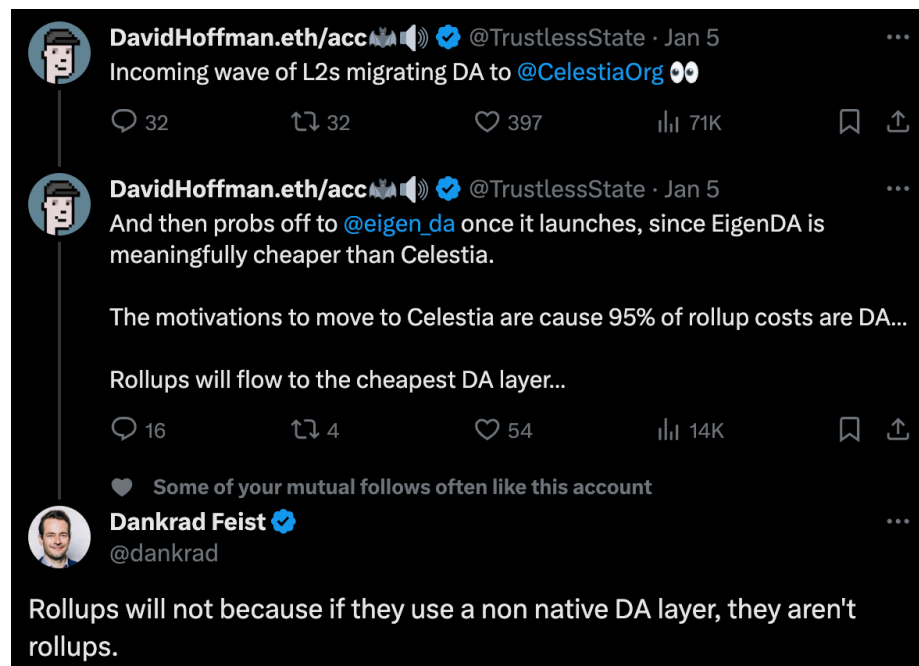


Figure 6. Part of the discussion about DA and L2 landscape by Dankrad Feist and David Hoffman

Source: x.com (January 5, 2024)

Currently, we identify the ensuing models as trust-reduced L2 solutions [6]:

1. **ZK (zero-knowledge) rollups:** These are advanced protocols that publish transaction data on L1, inheriting Ethereum's robust data availability and consensus mechanisms. They utilize intricate validity proofs to certify the veracity of the blockchain's state, offering a high-security profile while enhancing throughput. Notable projects include **zkSync Era**, **Linea**¹⁸, **Scroll**¹⁹, and **Starknet**²⁰.
2. **Optimistic rollups:** Functioning with a similar philosophy to ZK rollups, optimistic rollups preserve transaction data on L1. They rely on a system of fraud proofs to validate the state, with a main difference being the assumption that transactions are valid by

¹⁶ <https://celestia.org/>

¹⁷ <https://www.blog.eigenlayer.xyz/intro-to-eigenda-hyperscale-data-availability-for-rollups/>

¹⁸ <https://linea.build/>

¹⁹ <https://scroll.io/>

²⁰ <https://www.starknet.io/en>

default unless proven fraudulent. Notable projects include **Arbitrum One**, Optimism²¹, and **Base**.

3. **State Channels:** These require a level of vigilance from the users, who must actively maintain their transaction data. This is crucial for securing their ability to exit the system, limiting the technology's use to specific scenarios that can accommodate active user engagement. A good example of such a solution is Lightning Network²² on Bitcoin²³.
4. **Plasma:** This framework also refrains from publishing transaction data on-chain, employing fraud proofs for dispute resolution. Users are responsible for keeping their data to ensure proper exit mechanisms, and while Plasma operates with less user involvement than state channels, it does not facilitate general computation. Plasma chains are similar to Optimiums but offer permissionless exit [7].

With the introduction of further trust prerequisites, we can identify the subsequent classifications that fall outside the conventional boundaries of L2 architecture:

1. **Validiums:** Systems under this category depend on external data publication, utilizing validity proofs. They provide scalability by offloading data from Ethereum, yet they introduce trust assumptions related to the external data handling mechanisms [8]. Notable projects include StarkEx²⁴ based projects like Immutable X²⁵ or ApeX²⁶.
2. **Optimiums:** Known for their reliance on fraud proofs, these systems are similar to Optimistic Rollups but differ in that they publish their data externally, placing some trust outside of the Ethereum mainnet. Notable projects include Manta Pacific²⁷, Mantle²⁸ and Metis Andromeda²⁹. It is important to note that most of those projects are currently undergoing issues with their data availability and should be consider not fully safe.
3. **Side Chains:** Independent blockchains that run parallel to Ethereum and provide a separate environment for processing transactions and smart contracts. They have their own consensus mechanisms and are linked to Ethereum through two-way pegs. However, they introduce additional trust layers as they rely on their own security protocols, which are distinct from Ethereum's. Notable projects include Polygon PoS³⁰ that is currently undergoing its migration to Validium with Polygon PoS 2.0.

This paper concentrates on the two primary and most widely adopted L2 mechanisms - Optimistic and ZK rollups. It specifically examines three of the largest platforms (Figure 7.) - Arbitrum One, Base, and zkSync Era.

²¹ <https://www.optimism.io/>

²² <https://lightning.network/>

²³ <https://bitcoin.org/en/>

²⁴ <https://starkware.co/starkex/>

²⁵ <https://www.immutable.com/>

²⁶ <https://www.apex.exchange/>

²⁷ <https://pacific.manta.network/>

²⁸ <https://www.mantle.xyz/>

²⁹ <https://www.metis.io/>

³⁰ <https://polygon.technology/polygon-pos>

#	NAME	RISKS	TYPE	STAGE	PURPOSE	TOTAL	MKT SHARE
1	Arbitrum One		Optimistic Rollup	STAGE 1	Universal	\$11.06B + 10.45%	49.93%
2	OP Mainnet		Optimistic Rollup	STAGE 0	Universal	\$6.01B + 3.37%	27.14%
3	Metis		Optimism	n/a	Universal	\$963M + 40.19%	4.35%
4	Manta Pacific		Optimism	n/a	Universal	\$845M + 12.85%	3.81%
5	Base		Optimistic Rollup	STAGE 0	Universal	\$753M + 1.86%	3.40%
6	zkSync Era		ZK Rollup	STAGE 0	Universal	\$549M + 3.47%	2.48%

Figure 7. Six scaling solutions with the most market share
Source: l2beat.com (January 16, 2024)

2.3 Optimistic rollup vs ZK rollup

Rollups have been identified as a significant scaling strategy that offloads transaction processing to a secondary blockchain (Figure 8.), later consolidating these transactions into batches for submission on the primary blockchain, such as Ethereum. This approach lightens the burden on the main blockchain, accelerating transactions and reducing costs, while maintaining the security benefits of the primary Layer 1 chain.

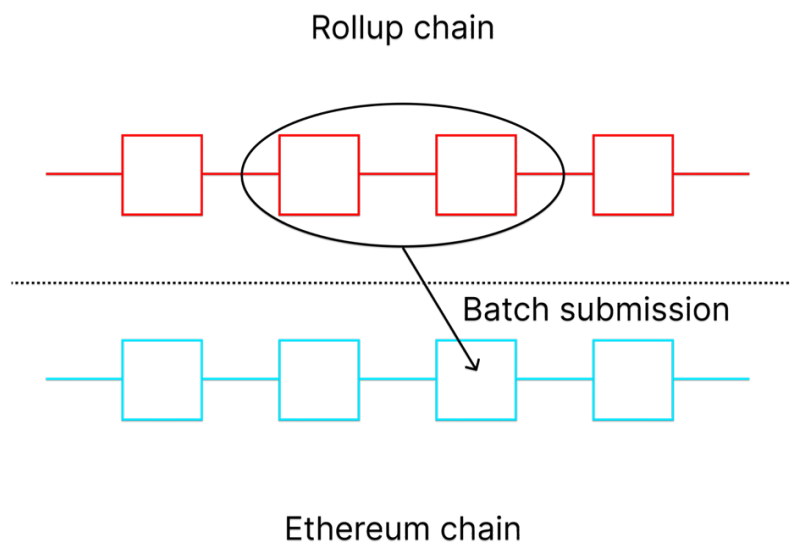


Figure 8. Visualization of the rollup transaction posting
Source: Author's own work

The core distinction between optimistic and ZK rollups lies in their verification methods [9]. Optimistic rollups operate on a presumption of honesty, assuming that the transaction batches submitted back to the main blockchain by validators are accurate. Nonetheless, during a subsequent waiting period, known as the challenge period, other network participants called verifiers have the opportunity to audit the batches and present fraud proofs if discrepancies are discovered [10]. Currently, leading L2 solutions like Arbitrum and employ an optimistic approach, partly due to its relative simplicity in implementation. This simplicity often stems

from the reliance on a single, centralized node or sequencer³¹ to process transactions, which enhances scalability. However, optimistic rollups face challenges, such as delays in transaction finality due to the challenge period, which can last up to a week, potentially leading to censorship and security concerns. Despite this, there is a movement towards the decentralization of sequencers. Furthermore, it has become apparent that developing effective fraud proof systems is more complex than previously thought, potentially giving ZK rollups an edge as the technology matures.

Conversely, ZK rollups do not take transaction validity for granted but instead use mathematical proofs to validate each transaction. This means that while solutions like zkSync Era require more computational resources, resulting in higher transaction costs than optimistic rollups, they benefit from quicker withdrawal times due to the absence of a challenge period. ZK rollups also enhance security and privacy through the use of zero-knowledge proofs, despite being more intricate and less compatible with Ethereum's smart contract capabilities.

In the long-term view, ZK rollups are seen by key industry figures, including Vitalik Buterin, as the ultimate evolution of rollup technology. Buterin has posited that ZK rollups will eventually outperform optimistic rollups, although he admits that ZK rollups are still developing and present more challenges for builders.

Market share analysis shows that optimistic rollups dominate (Figures 9. and 10.), holding almost 82% of the market according to L2BEAT, while ZK rollups hold a smaller portion (Figures 11. and 12.), around 8%. The market dominance of optimistic rollups is partly due to platforms like Arbitrum, which hosts the GMX³² decentralized exchange - a major player in decentralized finance - and Optimism, which has introduced the Superchain³³ concept and brought on significant partners like Coinbase with their Base solution.

Considering the TVL in these platforms, it's clear that although ZK rollups are gaining traction, optimistic rollups will likely remain a significant part of the ecosystem for several more years [11]. This reflects the current stage of development, where the balance between technological maturity and market adoption continues to evolve. It is equally critical to recognize, though, that the process of technically implementing a reliable fraud proof system has proven to be more time-intensive and resource-demanding than the establishment of a rudimentary ZK rollup.

³¹<https://www.obvious.technology/blogs/what-is-a-sequencer-and-why-is-it-important>

³² <https://gmx.io/>

³³ <https://app.optimism.io/superchain>

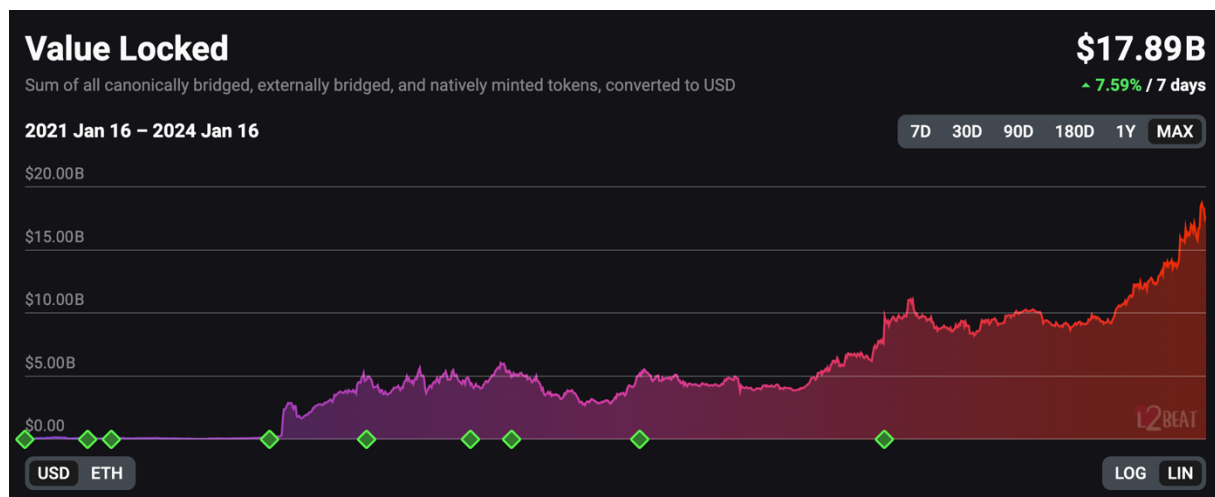


Figure 9. TVL growth of all optimistic rollups combined
Source: l2beat.com (January 16, 2024)

#	NAME	RISKS	TYPE	STAGE	PURPOSE	TOTAL	MKT SHARE
1	Arbitrum One		Optimistic Rollup	STAGE 1	Universal	\$11.06B ▲ 10.45%	49.93%
2	OP Mainnet		Optimistic Rollup	STAGE 0	Universal	\$6.01B ▲ 3.37%	27.14%
3	Base		Optimistic Rollup	STAGE 0	Universal	\$753M ▲ 1.86%	3.40%
4	Aevo		Optimistic Rollup	STAGE 0	DEX	\$22.73M ▲ 2.01%	0.10%
5	Boba		Optimistic Rollup	STAGE 0	Universal	\$16.01M ▲ 13.65%	0.07%
6	Zora		Optimistic Rollup	STAGE 0	Universal, NFT	\$11.34M ▲ 12.09%	0.05%
7	Lyra		Optimistic Rollup	STAGE 0	Exchange	\$5.26M ▲ 47.48%	0.02%
8	Kroma		Optimistic Rollup	STAGE 0	Universal	\$2.03M ▲ 66.19%	0.01%
9	PGN		Optimistic Rollup	STAGE 0	Universal	\$1.49M ▼ 0.22%	0.01%
10	Parallel		Optimistic Rollup	IN REVIEW	Universal, DeFi	\$39.14K ▲ 77.37%	0.00%
11	HoneyPot		Optimistic Rollup	STAGE 0	Bug bounty	\$23.03K ▲ 2.63%	0.00%
12	Fuel v1		Optimistic Rollup	STAGE 2	Payments	\$653 ▲ 8.89%	0.00%

Figure 10. All currently deployed optimistic rollups and their details
Source: l2beat.com (January 16, 2024)

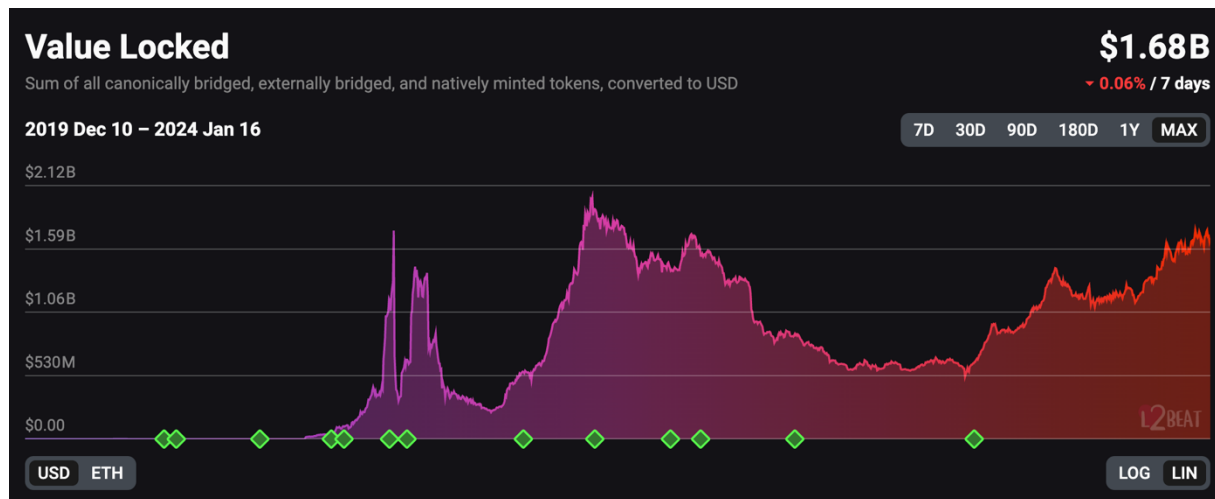


Figure 11. TVL growth of all ZK rollups combined
Source: l2beat.com (January 16, 2024)

#	NAME	RISKS	TYPE	STAGE	PURPOSE	TOTAL	MKT SHARE
1	zkSync Era		ZK Rollup	STAGE 0	Universal	\$549M ▼ 3.47%	2.48%
2	dYdX v3		ZK Rollup	STAGE 1	Exchange	\$336M ▼ 4.12%	1.52%
3	Linea		ZK Rollup	STAGE 0	Universal	\$201M ▲ 4.56%	0.91%
4	Starknet		ZK Rollup	STAGE 0	Universal	\$169M ▲ 10.87%	0.76%
5	Polygon zkEVM		ZK Rollup	STAGE 0	Universal	\$115M ▲ 2.22%	0.52%
6	Loopring		ZK Rollup	STAGE 0	NFT, AMM	\$112M ▲ 0.51%	0.51%
7	zkSync Lite		ZK Rollup	STAGE 1	Payments	\$76.11M ▲ 5.23%	0.34%
8	Scroll		ZK Rollup	STAGE 0	Universal	\$63.21M ▲ 9.21%	0.29%
9	DeGate V1		ZK Rollup	STAGE 2	Exchange	\$37.77M ▼ 10.78%	0.17%
10	ZKSpace		ZK Rollup	STAGE 0	NFT, AMM, Payments	\$6.35M ▼ 9.08%	0.03%
11	Paradex		ZK Rollup	STAGE 0	Exchange	\$5.02M ▲ 7.87%	0.02%

Figure 12. All currently deployed ZK rollups and their details
Source: l2beat.com (January 16, 2024)

3. Technical aspects of the analysis of selected solutions

This chapter provides a comprehensive review of three selected projects, assessing the potential risks of fund loss, their level of decentralization, and their upgrade details. Each project is evaluated impartially against the same criteria. It also delves into the theory of rollup stages, originally proposed by Vitalik Buterin [12] and subsequently refined by various entities. For this study, the L2BEAT frameworks will be employed for their detailed approach, refinement, and noteworthy impartiality.

3.1 Risk analysis

When deciding about sending your assets to a certain rollup via a bridge, it is important to consider 5 important questions:

1. How is the validity of the system state checked?
2. Is the data needed to reconstruct the state available?
3. How much time do users have to exit the system in case of an unwanted upgrade?
4. What happens if the sequencer, an entity responsible for constructing blocks and deciding on the ordering of transactions is offline or censors individual user?
5. What happens if the proposer³⁴, an entity responsible for submitting state commitments to Ethereum is offline?

Answers to those questions require checking the deployed code of the projects and analyzing it. It is not an easy task since smart contracts need to be verified first before anybody can see their source code. One of L2BEAT's goals is to ensure that projects will do it and then gather such pieces of information on their website for everybody to easily find and use. Based on that, we can prepare answers for each question for each of our evaluated projects.

3.2 Decentralization stage

The initial development phase of rollups often includes a period of centralized oversight, a phase sometimes described as the system wearing "training wheels," which facilitates updates and troubleshooting in a manageable setting. While such centralized control is a practical necessity in the beginning, the goal is for rollups to eventually stand independently, fully leveraging the security features of the L1 to which they post data.

To structure this evolution, L2BEAT created a framework [13], taking cues from milestones suggested by Vitalik Buterin, that organizes rollups into three progressive stages based on their dependency on centralized control mechanisms:

- **Initial Phase, Stage 0:** full training wheels — At this foundational level, rollup operators have significant control, yet open-source software is available, enabling state reconstruction from L1 data for state root verification.
- **Intermediate Phase, Stage 1:** limited training wheels — Here, the rollup begins to shift towards smart contract-based governance. A temporary Security Council³⁵ may persist to manage bugs. Key features include a working proof system, decentralized fraud proof

³⁴ <https://www.alchemy.com/overviews/proposer-builder-separation>

³⁵ <https://www.cryptofrens.info/p/is-a-rollup-just-a-multisig>

submissions, and user exits without operator intervention. The Security Council consists of various stakeholders acting as a provisional safety measure, albeit with inherent risks due to its retained authority.

- **Advanced Phase, Stage 2:** no training wheels — In the ultimate phase, the rollup is entirely governed by smart contracts, with a permissionless fraud proof system in place. Users have sufficient time to exit autonomously during non-consensual upgrades. The Security Council's duties are limited to on-chain resolvable soundness issues, safeguarding users from governance-related vulnerabilities.

3.3 Upgradeability

The avenues for code upgradeability within a project are critical – even if it meets all other criteria, a single malevolent change could lead to the loss of assets for all users. Therefore, data from L2BEAT's Upgradeability Report [14] was utilized. In this section, there will be references to specific elements of code or functions, but for a complete understanding of the upgrade architecture, the author refers to the report – as the topic is too extensive.

3.4 Arbitrum One analysis

3.4.1 Introduction to Arbitrum One

Arbitrum One [15] has emerged as a pioneering universal Optimistic rollup (Figure 13.) in the L2 ecosystem, developed by Offchain Labs [16] and administered by a decentralized autonomous organization (DAO), the Arbitrum DAO³⁶. With a TVL of 11.06 billion USD (Figure 14.) and an average of daily 9 TPS in the last year (Figure 15.), Arbitrum One commands a substantial presence, boasting nearly half (49.93%) of the entire L2 market share. This chapter evaluates Arbitrum One, applying the L2BEAT frameworks to scrutinize its risk factors, degree of decentralization, and upgrade mechanisms.

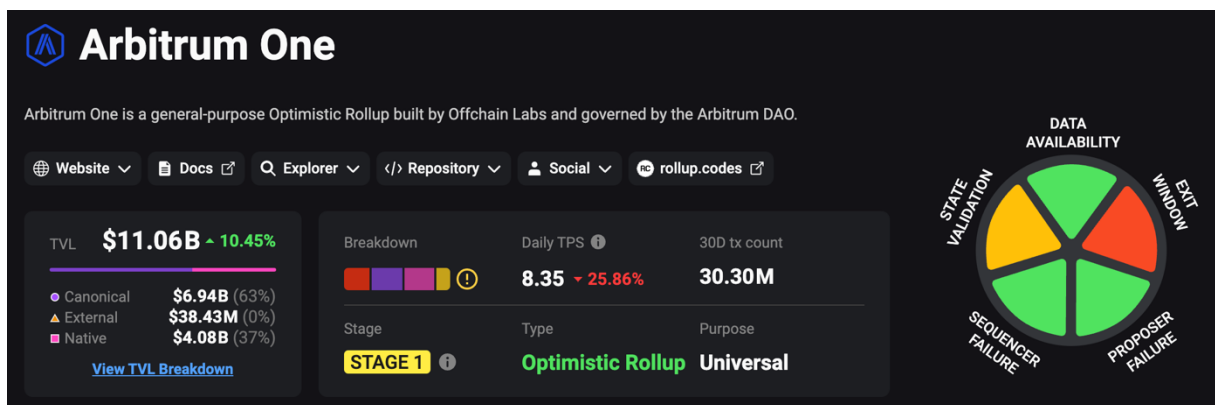


Figure 13. Arbitrum One details and summary
Source: l2beat.com (January 16, 2024)

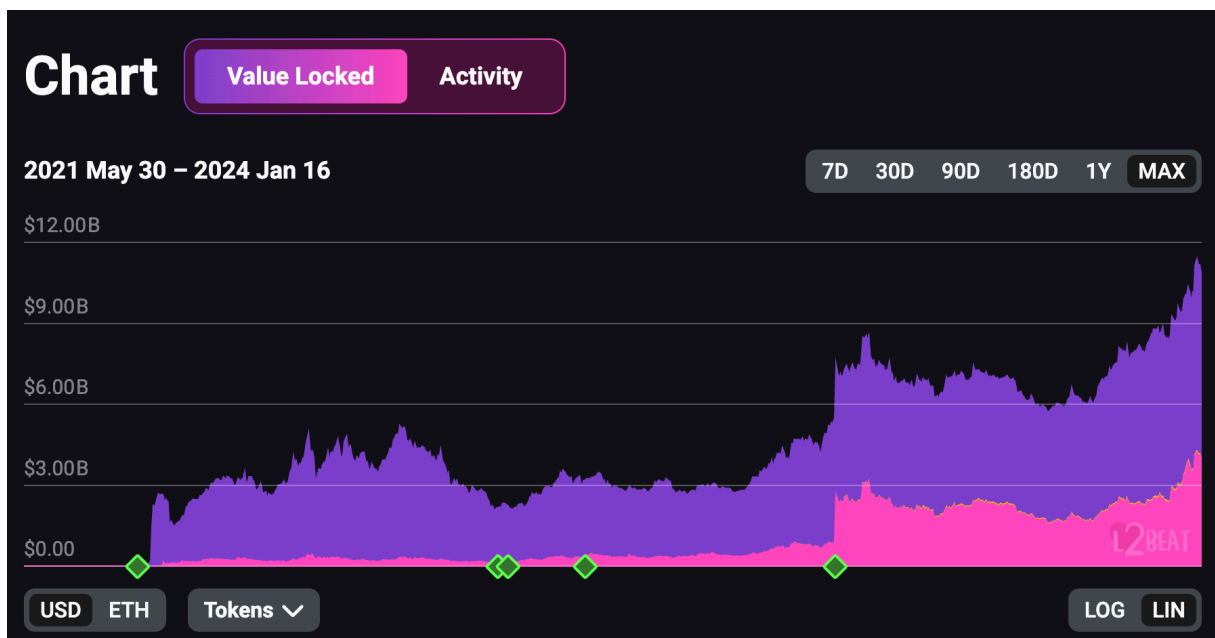


Figure 14. Arbitrum One TVL growth
Source: l2beat.com (January 16, 2024)

³⁶ <https://www.tally.xyz/gov/arbitrum>

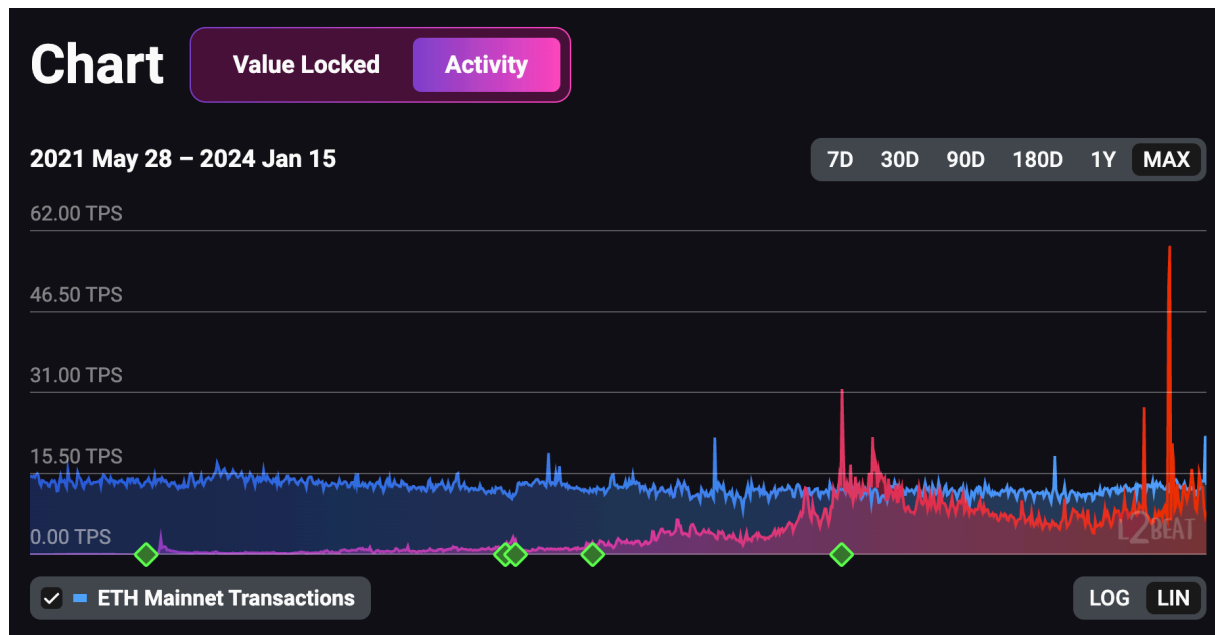


Figure 15. Arbitrum One activity growth compared to Ethereum
Source: l2beat.com (January 16, 2024)

3.4.2 Arbitrum One risk analysis

Arbitrum One ensures security across three key areas, possesses one minor risk, and faces one critical risk (Figure 16.):

1. Arbitrum One has implemented a fraud proof system, although it currently maintains a whitelist for validators, indicating a move toward but not fully achieving permissionless validation. Fraud proofs enable approved participants to challenge the state's accuracy through interactive proofs, necessitating a series of transactions over time. However, this process can be vulnerable to delay tactics within the challenge protocol.
2. The project ensures complete transparency by publishing all necessary data on-chain, essential for constructing proofs and verifying state transitions.
3. The exit window for users is two days, with upgrades initiated on L2 undergoing a three-day delay. Should there be a dispute, users have a limited time frame to execute an exit transaction before being forced to self-propose a new state root, followed by a waiting period of over six days for challenges, coinciding with the upgrade confirmation timeline.
4. Arbitrum One has mechanisms in place for sequencer failure, allowing users to self-sequence transactions by submitting them to L1 with a one-day delay. Additionally, in the absence of activity from whitelisted Proposers, any participant can assume the proposer role after a specific inactivity period, promoting continuity and resilience.

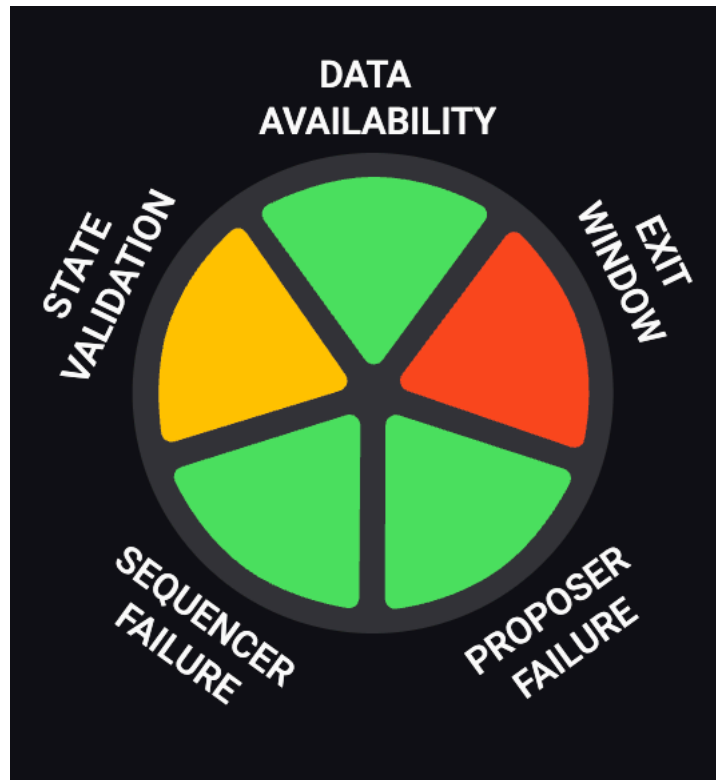


Figure 16. Arbitrum One risk analysis visualization
Source: l2beat.com (January 16, 2024)

3.4.3 Arbitrum One decentralization assessment

Arbitrum One is currently classified as Stage 1 in the decentralization lifecycle (Table 1.). This stage reflects a shift towards decentralized governance, although temporary centralized elements persist for oversight and management.

1. Arbitrum One satisfies all foundational requirements of Stage 0, affirming its identity as a rollup and ensuring the availability of state roots and input data on Ethereum L1. The project maintains an open-source node capable of reconstructing the state from L1 data, ensuring transparency and verifiability.
2. The platform has effectively deployed a complete proof system and has at least five external actors authorized to submit fraud proofs, signifying a move towards a more decentralized governance model. Users can independently exit the system without needing permissioned operators, and in the event of an unendorsed upgrade, they are afforded a minimum of seven days to exit, safeguarding their assets.
3. To reach Stage 2, Arbitrum One must address certain areas. Currently, the submission of fraud proofs is limited to a select group of actors, thus restricting it from becoming fully permissionless. Additionally, the protocol needs to extend the exit period for non-consensual upgrades to at least 30 days to ensure ample user protection. Finally, the actions of the Security Council must be confined strictly to resolving on-chain provable issues to prevent governance-related vulnerabilities.

Table 1. Decentralization assessment of Arbitrum One
Source: Author's own work based on data from l2beat.com

Requirement	Stage	Fulfillment
The project calls itself a rollup	Stage 0	Yes
L2 state roots are posted to Ethereum L1	Stage 0	Yes
Inputs for the state transition function are posted to L1	Stage 0	Yes
A source-available node exists that can recreate the state from L1 data	Stage 0	Yes
A complete and functional proof system is deployed	Stage 1	Yes
There are at least 5 external actors who can submit fraud proofs	Stage 1	Yes
Users are able to exit without the help of the permissioned operators	Stage 1	Yes
In case of an unwanted upgrade by actors more centralized than a Security Council, users have at least seven days to exit	Stage 1	Yes
The Security Council is properly set up	Stage 1	Yes
Fraud proof is system permissionless	Stage 2	No
Users have at least 30 days to exit in case of unwanted upgrades	Stage 2	No
Security Council is restricted to act only due to errors detected on chain	Stage 2	No

3.4.4 Arbitrum One upgrade pathways

There are three distinct pathways (Figure 17.) for executing upgrades within the Arbitrum systems:

- **DAO-Initiated Upgrades:** This primary path commences with a proposal to the *CoreGovernor*, triggering the voting process. Successful proposals, post quorum achievement, are queued in the L2 *Timelock*. A waiting period is observed before an L2 to L1 message is sent. The *Security Council* has the authority to cancel proposals during this period. Subsequently, the proposal endures a challenge period and another waiting phase in the L1 *Timelock* before the *UpgradeExecutor* enacts the changes.

- **Security Council Fast-Track Upgrades:** For minor, non-contentious upgrades, the Security Council can fast-track changes on L2, a provision made to counteract the potential sluggishness of DAO processes (*ARB³⁷ Holders*). These upgrades undergo the same procedural steps as DAO-initiated upgrades but are started by a lowered threshold of approval within the *Security Council*.
- **Emergency Upgrades:** In exigent circumstances, the *Security Council* can directly command the *UpgradeExecutor* to implement immediate upgrades without the standard delays, a measure intended for critical, time-sensitive situations.

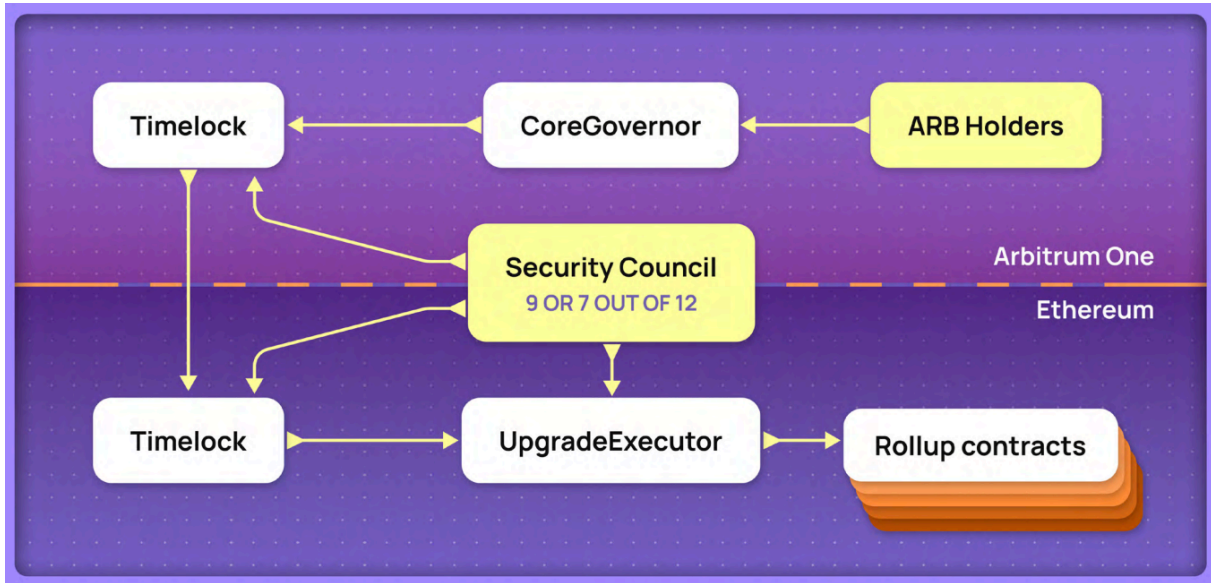


Figure 17. Upgrade system architecture of Arbitrum One
Source: L2BEAT report - Upgradeability of Ethereum L2s [14]

3.5 Base analysis

3.5.1 Introduction to Base

Base is a Layer 2 universal purpose optimistic rollup developed by Coinbase³⁸, utilizing the Optimism OP Stack technology as a component of the Superchain initiative [17]. It is the newest project among those evaluated in this study. With a TVL of 754 million USD (Figure 19.), and an average of daily 2.4 TPS since its launch (Figure 20.), Base has quickly asserted itself in the competitive Layer 2 market, achieving around 3.40% market share. It is important to note that analysis of this project can be also used in reference to other Superchain members like Optimism or Zora.

³⁷ <https://docs.arbitrum.foundation/concepts/arb-token>

³⁸ <https://www.coinbase.com/>

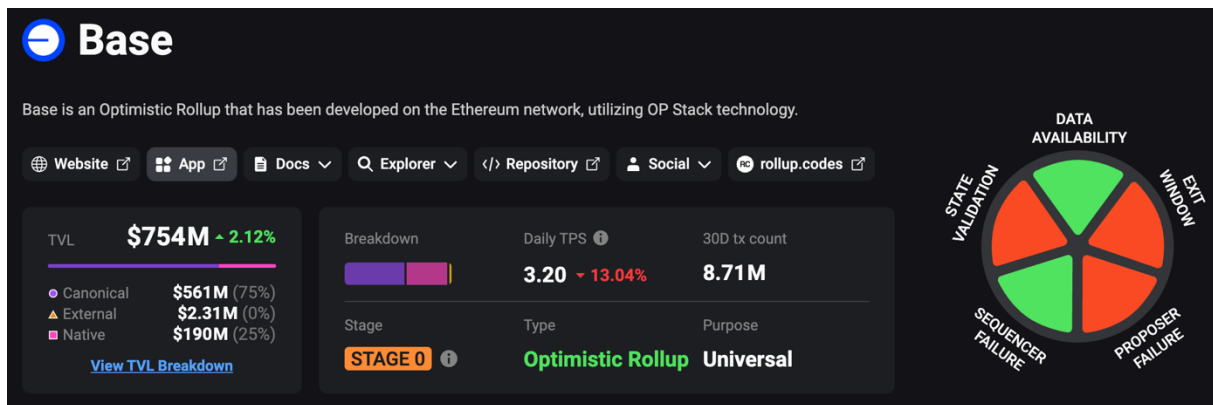


Figure 18. Base details and summary
Source: l2beat.com (January 16, 2024)

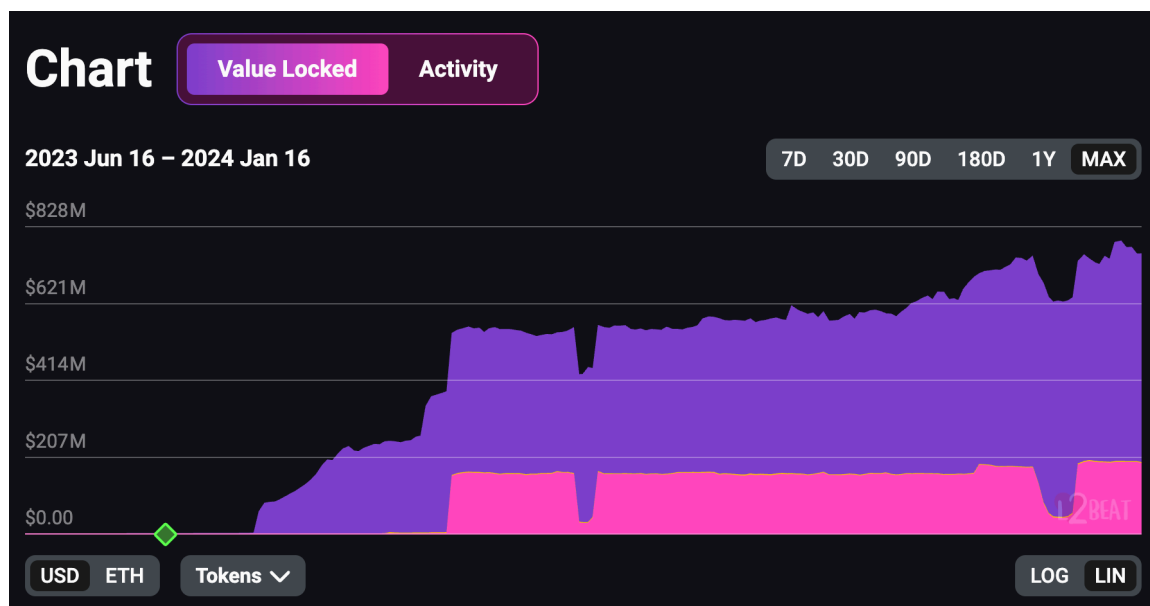


Figure 19. Base TVL growth
Source: l2beat.com (January 16, 2024)

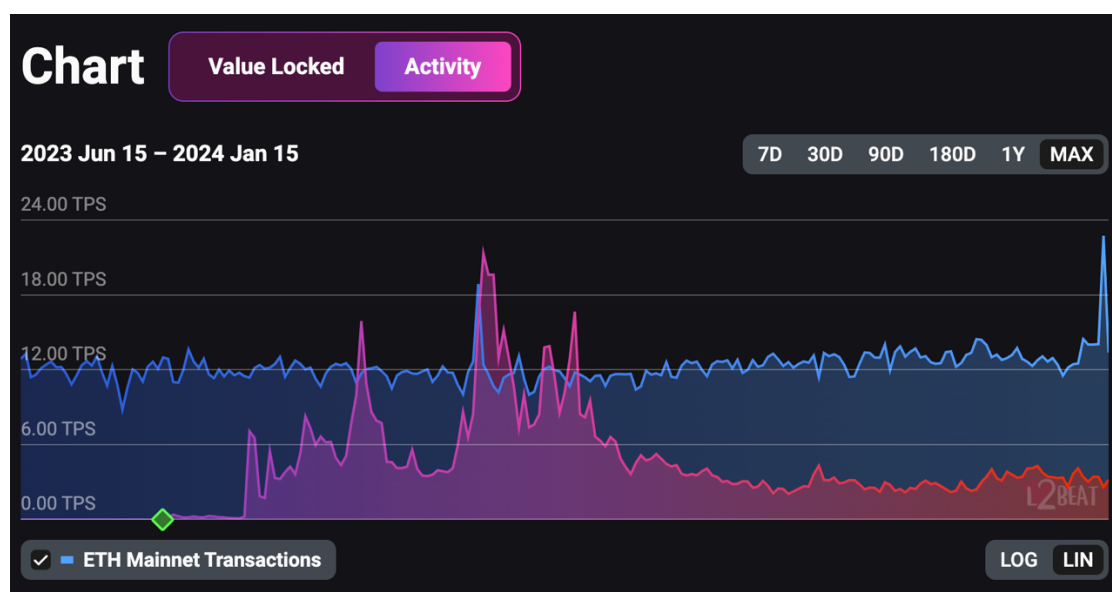


Figure 20. Base activity growth compared to Ethereum
Source: l2beat.com (January 16, 2024)

3.5.2 Base risk analysis

Base ensures security across two key areas and faces three critical risks (Figure 21.):

1. The fraud proof system, a key security feature for Layer 2 solutions like Base, is currently in development. At this stage, users are required to place trust in the block proposer's honesty in submitting correct L1 state roots. While this trust-based model is in place, there is an acknowledged risk of invalid state roots being permitted within the system.
2. Base maintains transparency and data availability by publishing all requisite data on-chain, which is essential for constructing proofs and verifying state transitions.
3. One significant concern with Base is the absence of an exit window for users in the event of an unwanted upgrade. The contracts within Base are configured for instant upgradability, which can potentially leave users without a predefined time frame to withdraw their funds if they disagree with changes made to the system.
4. Base provides a mechanism for transaction continuity in the event of sequencer failure. Users have the ability to self-sequence transactions by sending them directly to L1, although this process includes a 12-hour delay.
5. In the current state, if the whitelisted proposer fails to publish state roots on L1, users are unable to withdraw their funds, effectively freezing their assets within the system. This highlights a centralization risk, as the ability to publish state roots is limited to a select group of whitelisted proposers.

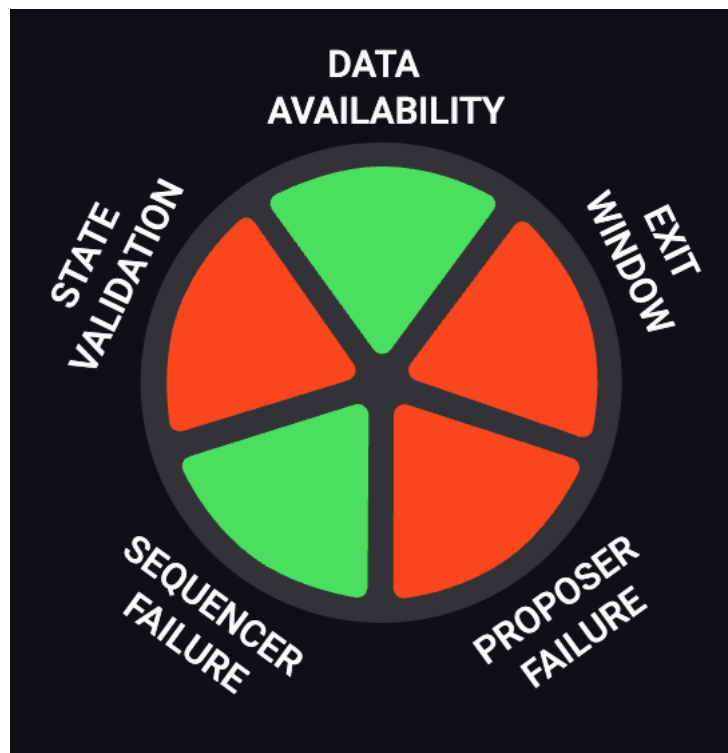


Figure 21. Base risk analysis visualization
Source: l2beat.com (January 16, 2024)

3.5.3 Base decentralization assessment

In its current iteration, Base is positioned at Stage 0 (Table 2.) within the decentralization framework of optimistic rollups.

1. It fulfills the baseline criteria for this category, identifying as a rollup and maintaining the necessary linkage between Layer 2 and Ethereum Layer 1 by posting state roots and state transition inputs. Additionally, the infrastructure of Base is reinforced by the availability of source-accessible nodes that enable the reconstruction of state from Layer 1 data, a critical component for maintaining the integrity and transparency of the network.
2. Progressing towards Stage 1, Base faces several challenges that need to be addressed to enhance its decentralization. The completion of a robust proof system is still underway, which is essential for eliminating trust in centralized block proposers. Moreover, the current architecture allows permissioned operators the ability to censor user withdrawals, a practice that stands in contrast to the ethos of decentralization. Furthermore, the existing upgrade protocol allows for changes to be implemented with a degree of central control that provides users with an insufficient timeframe to exit in case of undesired amendments by the operators.
3. For Base to evolve to Stage 2, the pivotal concern lies in the upgrade policy. The protocol should be amended to offer a minimum of 30 days for users to exit following upgrades that are not linked to on-chain verifiable bugs, thereby granting users ample time to respond to and protect against potential system-wide changes that they do not consent to.

Table 2. Decentralization assessment of Base
Source: Author's own work based on data from l2beat.com

Requirement	Stage	Fulfillment
The project calls itself a rollup	Stage 0	Yes
L2 state roots are posted to Ethereum L1	Stage 0	Yes
Inputs for the state transition function are posted to L1	Stage 0	Yes
A source-available node exists that can recreate the state from L1 data	Stage 0	Yes
A complete and functional proof system is deployed	Stage 1	No
There are at least 5 external actors who can submit fraud proofs	Stage 1	No
Users are able to exit without the help of the permissioned operators	Stage 1	No

In case of an unwanted upgrade by actors more centralized than a Security Council, users have at least seven days to exit	Stage 1	No
The Security Council is properly set up	Stage 1	No
Fraud proof is system permissionless	Stage 2	No
Users have at least 30 days to exit in case of unwanted upgrades	Stage 2	No
Security Council is restricted to act only due to errors detected on chain	Stage 2	No

3.5.4 Base upgrade pathways

The following methods can be used to control and modify Base (Figure 22.):

- **Direct Contract Upgrades:** These can be performed instantaneously by the *Base (Optimism) Multisig* (Multi-Signature Wallet³⁹), allowing comprehensive changes to rollup mechanisms. This power includes the ability to modify contracts potentially affecting user funds.
- **Withdrawal Halt:** The *Guardian* role, currently occupied by the *Base Multisig*, has the capacity to stop withdrawals as a form of control or response to threats.
- **Output Roots Management:** The *Challenger*, also held by the *Base Multisig*, can delete non-finalized output roots, affecting the state root confirmation process.

The above visualization from the report is dedicated to Optimism, but since Base is its direct copy, we can still use it as a reference.

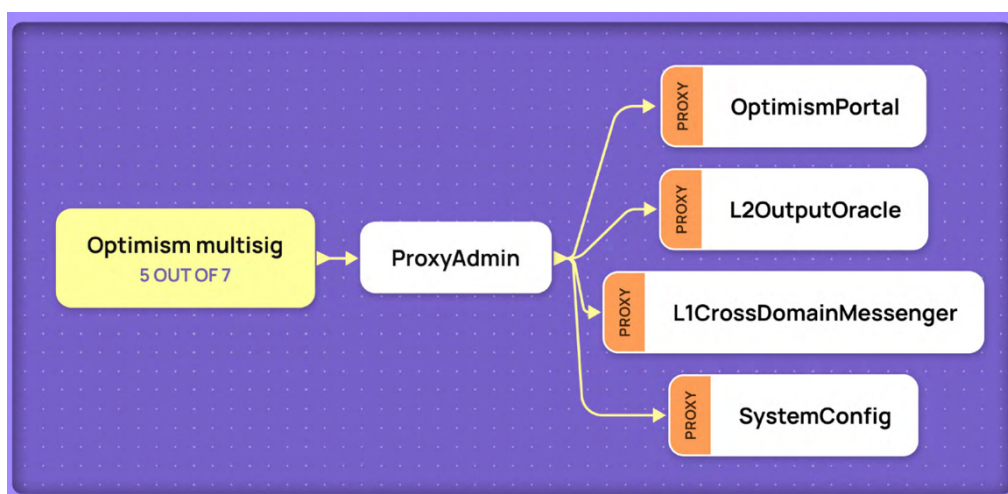


Figure 22. Upgrade system architecture of Base (Optimism)
Source: L2BEAT report - Upgradeability of Ethereum L2s [14]

³⁹ <https://coinmarketcap.com/academy/glossary/multi-signature-multi-sig>

3.6 zkSync Era analysis

3.6.1 Introduction to zkSync Era

zkSync Era [18] is a universal purpose ZK rollup solution (Figure 23.) developed by Matter Labs⁴⁰, aiming to provide scalability to the Ethereum network while maintaining EVM (Ethereum Virtual Machine⁴¹) compatibility. Launched for general use in February 2023 after an initial phase of allowlisted access, zkSync Era is the successor to the earlier iteration known as zkSync Lite⁴². It holds a TVL of approximately 549 million USD (Figure 24.), accounting for 2.48% of the market share within the Layer 2 ecosystem and had a daily average of 7.32 TPS (Figure 25.) since its launch.

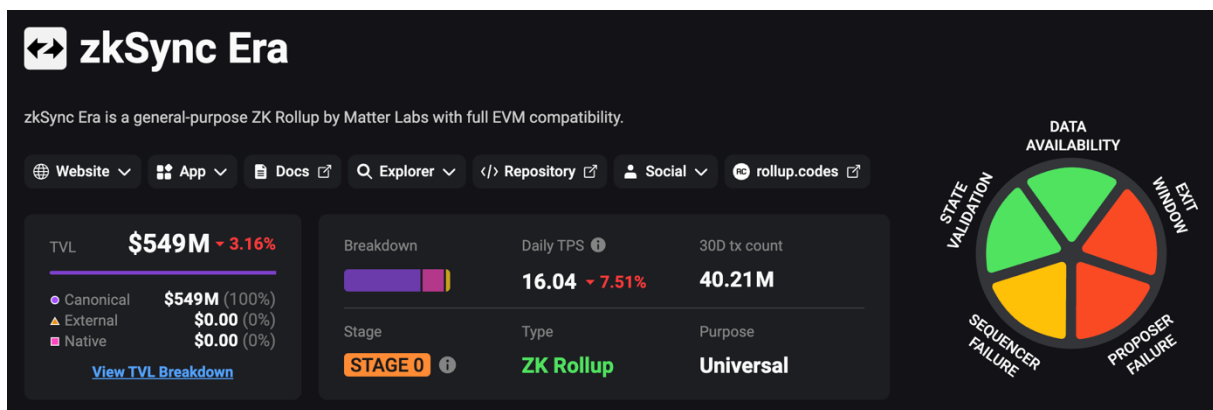


Figure 23. zkSync Era details and summary
Source: l2beat.com (January 16, 2024)

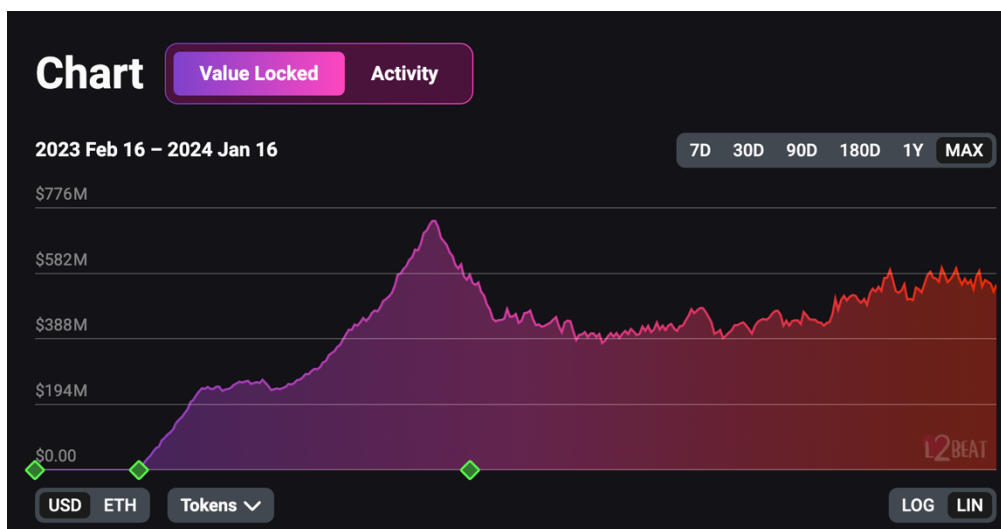


Figure 24. zkSync Era TVL growth
Source: l2beat.com (January 16, 2024)

⁴⁰ <https://matter-labs.io/>

⁴¹ <https://ethereum.org/en/developers/docs/evm>

⁴² <https://lite.zksync.io/>

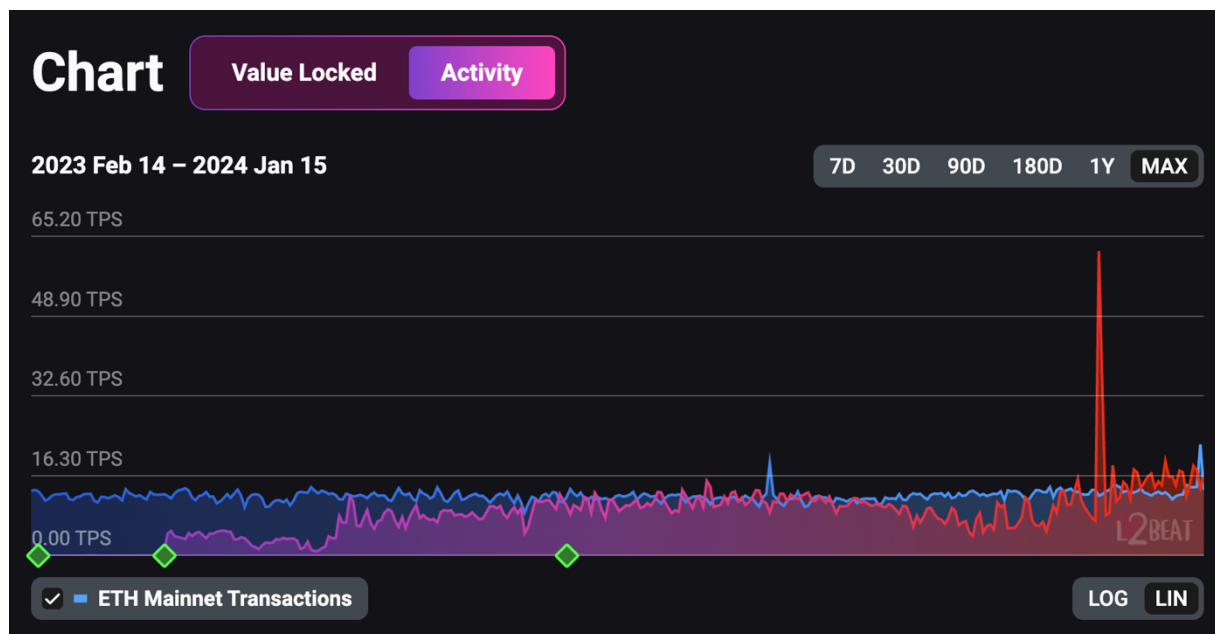


Figure 25. zkSync Era activity growth compared to Ethereum
Source: l2beat.com (January 16, 2024)

3.6.2 zkSync Era risk analysis

zkSync Era ensures security across two key areas, possesses one minor risk, and faces two critical risks (Figure 26.):

1. The system implements a withdrawal delay of 21 hours, which is subject to modification by a MultiSig wallet. This design choice introduces a layer of control over fund movement, potentially impacting liquidity and user experience.
2. zkSync Era employs the PLONK (Permutations over Lagrange-bases for Oecumenical Noninteractive arguments of Knowledge⁴³) zero-knowledge proof system with KZG (Kate Zaverucha Goldberg⁴⁴) commitments for state validation. This method is a cornerstone of its security, providing assurances about the correctness of state transitions without revealing underlying data.
3. All necessary data for proof construction, specifically state differences, are available on-chain.
4. A notable concern for zkSync Era users is the absence of a designated exit window in the event of an unwanted upgrade. The contracts within the platform are structured for instant upgradability, potentially leaving users with limited options to withdraw their funds during such changes.
5. In scenarios of sequencer failure, users have the option to submit transactions to a L1 queue. However, they lack the capability to force these transactions into the system. The sequencer, while unable to selectively bypass transactions, can halt the processing of the queue entirely, leading to a network-wide standstill.

⁴³ <https://arc.net/l/quote/ofexjogc>

⁴⁴ <https://arc.net/l/quote/bltvmsqm>

6. Regarding proposer failure, the system's design limits the publication of state roots on L1 to whitelisted proposers. Consequently, the failure of these proposers can result in the freezing of withdrawals, introducing a centralization risk and potential disruption for users seeking to access their funds.

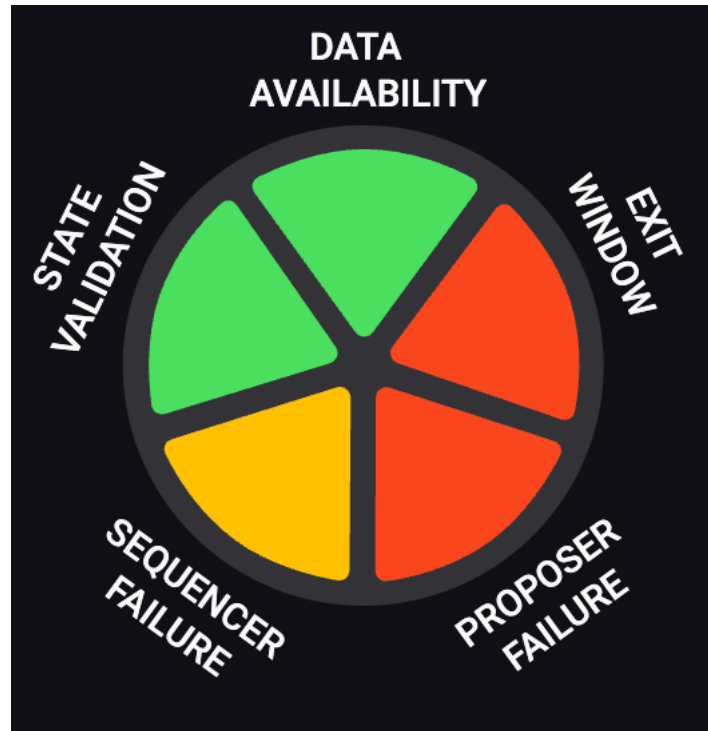


Figure 26. zkSync Era risk analysis visualization
Source: l2beat.com (January 16, 2024)

3.6.3 zkSync Era decentralization assessment

zkSync Era, as a Stage 0 ZK rollup, aligns with foundational requirements that define its classification within the decentralization framework (Table 3.).

1. These requirements include its self-identification as a rollup, the consistent posting of Layer 2 state roots to Ethereum Layer 1, and the availability of inputs for state transition functions on L1. Additionally, the presence of a source-available node capable of recreating the state from L1 data underscores its commitment to transparency and verifiability.
2. Moving toward Stage 1, zkSync Era has successfully deployed a complete and functional proof system, addressing one of the key requirements for this level. However, the platform still encounters challenges that need to be resolved. One significant issue is the potential for censorship of user withdrawals by the permissioned operators, a concern that points towards centralized control. Another aspect requiring attention is the upgrade process; as it currently stands, the system provides less than seven days for users to exit in scenarios where the permissioned operator is either down or engaging in censorship activities.

3. To achieve Stage 2 decentralization, zkSync Era is tasked with extending the exit period for upgrades that are unrelated to on-chain provable bugs. Presently, the system does not offer the recommended 30-day window for user exits in these circumstances, which is essential for safeguarding user rights and maintaining trust during significant changes to the system.

Table 3. Decentralization assessment of zkSync Era
Source: Author's own work based on data from l2beat.com

Requirement	Stage	Fulfillment
The project calls itself a rollup	Stage 0	Yes
L2 state roots are posted to Ethereum L1	Stage 0	Yes
Inputs for the state transition function are posted to L1	Stage 0	Yes
A source-available node exists that can recreate the state from L1 data	Stage 0	Yes
A complete and functional proof system is deployed	Stage 1	Yes
Users are able to exit without the help of the permissioned operators	Stage 1	No
In case of an unwanted upgrade by actors more centralized than a Security Council, users have at least seven days to exit	Stage 1	No
Users have at least 30 days to exit in case of unwanted upgrades	Stage 2	No

3.6.4 zkSync Era upgrade pathways

zkSync Era, in its current configuration, has designated two key roles with specific permissions within its system: the *Governor* and the *Security Council*. The *Governor* role is operated through the team's *MultiSig*, whereas the *Security Council* is currently unassigned, set to a null address. The upgrade process of zkSync Era is intricately managed by the *DiamondCut* facet within the zkSync contract (Figure 27.). This process includes two distinct types of upgrades: transparent upgrades, where the complete upgrade code is provided openly, and shadow upgrades, which are more discreet, revealing only the hash of the upgrade code.

The protocol for executing upgrades involves several steps:

1. Initiation: The process starts with the *Governor* proposing an upgrade, which can be either transparent or shadow.

2. **Approval:** The *Security Council* plays a role in this phase, where they have the discretion to approve the proposed upgrade. This approval is not a prerequisite for transparent upgrades.
3. **Execution:** Once the *Security Council* gives its approval, the *Governor* is empowered to implement the upgrade immediately. In cases where the *Security Council* does not approve, there is a predetermined notice period, which currently stands at zero, that must pass before executing transparent upgrades.
4. **Revocation:** There is a provision for the *Governor* to cancel any proposed upgrade that has not yet been implemented.

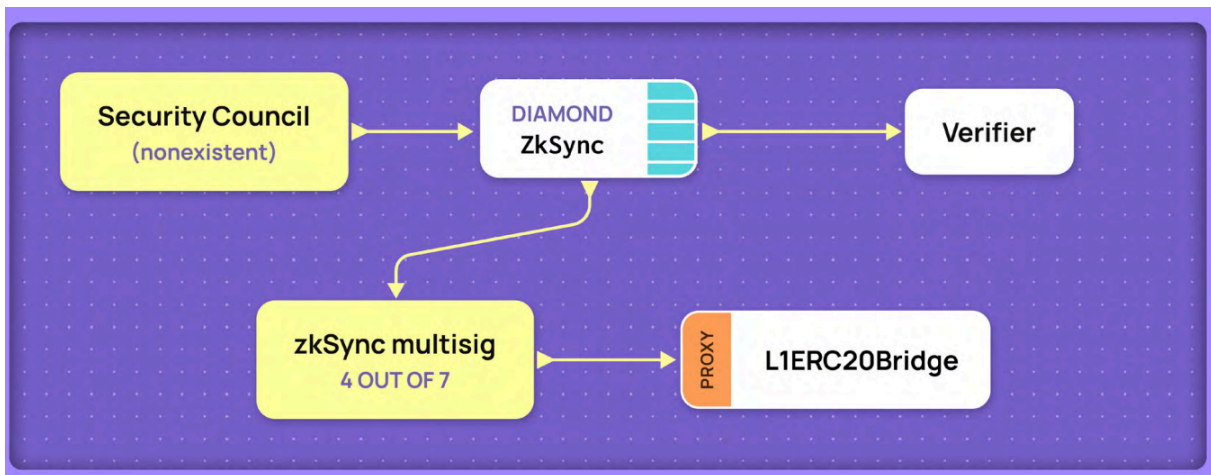


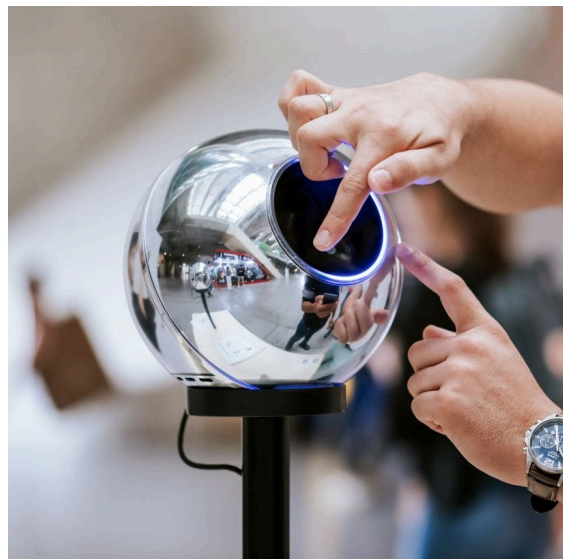
Figure 27. Upgrade system architecture of zkSync Era
Source: L2BEAT report - Upgradeability of Ethereum L2s [14]

4. A brief introduction to additional resources to consider

4.1 Worldcoin – building on L2 instead of L1

In the subsequent chapter of this engineering thesis, we will briefly consider Worldcoin as an example of a project built on a Layer 2 solution [19], specifically Optimism (an original iteration of every Superchain project). This discussion does not delve deeply into the intricacies of Worldcoin; instead, it serves as an illustrative example, setting the stage for further research.

Worldcoin is an initiative spearheaded by Sam Altman⁴⁵ and Alex Blania⁴⁶. Its mission is to ensure that in the impending AI revolution, humans can be distinguished from advanced bots on the internet. The founders believe this can be achieved by putting each individual through iris scanning by a device called an orb (Figure 28.) and recording the corresponding hash of that scan on the blockchain which later can be used to verify the humanity of such users. Subsequently, users receive a periodic allocation of WLD⁴⁷ tokens every two weeks as a form of reward.



*Figure 28. A photo of an orb, a device used to scan the iris
Source: Worldcoin's official Instagram account*

This project stands as a potent method to draw more users into the Web3 domain, with user numbers expected to continually rise until a significant portion of the population has been scanned. To date, nearly 3 million individuals have been scanned (Figure 29.). The volume of user transactions, such as sending tokens or exchanging them, has reached approximately 14 million. Following this logic, if the project indeed remains in use by hundreds of millions of

⁴⁵ <https://blog.samaltman.com/>

⁴⁶ <https://www.crunchbase.com/person/alex-blania>

⁴⁷ <https://worldcoin.org/worldcoin-token>

people, a network like Ethereum would be unable to handle the sheer volume of transactions. Worldcoin alone could lead to network congestion and a spike in gas prices to exorbitant levels.

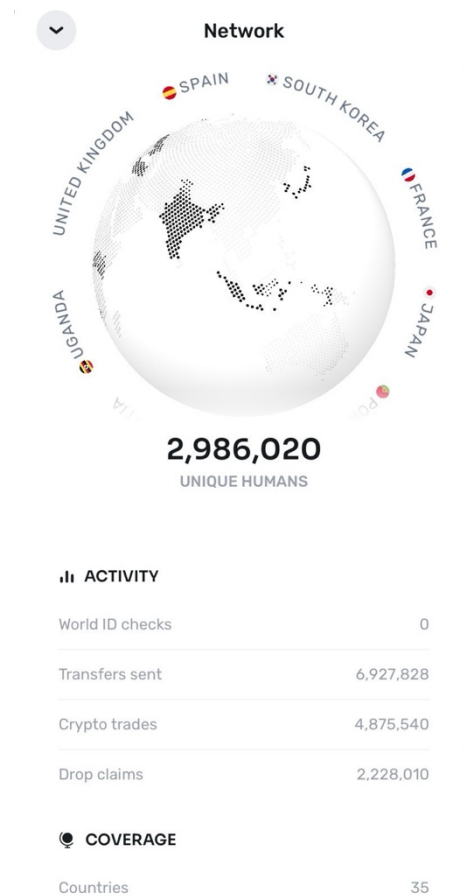


Figure 29. Details about the activity of Worldcoin users
Source: World App (January 9, 2024)

Therefore, the Worldcoin team has selected the Optimism network for its infrastructure. This choice allows for unfettered growth over the next several years without concerns over clogging the underlying infrastructure. Given that the project focuses on sensitive topics like privacy and proof of humanity, the chosen blockchain must offer sufficient decentralization and security, attributes that Ethereum is known for. The Optimism rollup is an ideal solution in this regard. However, it is evident that if the project were to be adopted for government applications or private businesses, it would eventually require its own dedicated L2 solution like a custom rollup, and subsequently, an even more robust system capable of scaling Ethereum's transaction capacity not just manifold, but by hundreds or thousands of times.

4.2 EIP-4844

Another factor that the author of the paper identifies as important is the forthcoming EIP-4844⁴⁸ upgrade to the Ethereum code, known as proto-danksharding [20]. This topic is quite broad, so it will only be briefly mentioned here to ensure that it is acknowledged in the

⁴⁸ <https://coinmarketcap.com/academy/glossary/ethereum-improvement-proposal-eip>

final evaluation. This change is likely to be deployed on the mainnet shortly after this paper is completed.

Proto-danksharding, also referred to as EIP-4844, is a precursor to the full danksharding framework proposed for Ethereum. It diverges significantly from prior sharding concepts⁴⁹, focusing more on data storage enhancement than transaction space expansion. This approach aligns with Ethereum's strategy to prioritize rollup technologies, emphasizing data-centric solutions over traditional transaction-centric models.

A key innovation in proto-danksharding is the introduction of blob-carrying transactions. These transactions differ from standard ones by including a large data section, known as a blob. Each blob, roughly 125 KB in size, offers a cost-efficient alternative to calldata. The blobs are designed to be recognized by the Ethereum network in terms of their presence but are not directly accessible for execution by the EVM, which only acknowledges a commitment to the blob.

An important feature of proto-danksharding is its approach to data availability and pruning. The system ensures that blobs are available on the network for about two weeks. This duration balances the need for L2 access with the practicality of disk space management. Unlike calldata, which is stored indefinitely, this temporary storage approach of blobs allows for more economical data handling.

In essence, proto-danksharding is a strategic step towards implementing danksharding in Ethereum. It introduces critical changes in transaction structure and data management, primarily to support rollup technologies. These modifications are expected to reduce transaction fees significantly, making Ethereum more scalable and user-friendly. Proto-danksharding thus plays a pivotal role in Ethereum's L2 ecosystem growth.

⁴⁹ <https://101blockchains.com/ethereum-sharding-explained/>

5. Design of tools for data collection

In this chapter, we embark on a detailed exposition of the methodologies and instruments deployed for data collection within the context of this study. Our primary objective is to sculpt a data collection framework that serves to impartially capture and process information. This framework will be the bedrock upon which we base our evaluation of the performance metrics, cost implications, and operational efficiencies associated with Ethereum's main network and its ancillary L2 solutions.

The design of the data collection framework was meticulously crafted, with careful consideration given to the selection criteria to ensure that the data points gathered are relevant, accurate, and comprehensive. The data points of interest were strategically chosen to include key performance indicators such as transaction latency, cost per transaction, and throughput, which are critical in gauging the efficacy of blockchain solutions. The data analysis methods were pre-selected to include both quantitative and qualitative techniques, ensuring a balanced and multifaceted approach to the study.

To ensure the integrity and effectiveness of the data collection process, a robust technological stack has been employed. The simulation tool central to our study was developed using Node.js, a choice motivated by its non-blocking, event-driven architecture which allows for efficient data handling and network requests - characteristics particularly suited to blockchain transaction simulations. The use of Node.js also facilitated a seamless integration with the EVM enabling the tool to replicate the execution environment of the Ethereum network with high fidelity.

Within the Node.js environment, the Ethers.js library was utilized as a comprehensive gateway to the Ethereum blockchain. This library provided a set of tools that allowed for the creation, signing, and broadcasting of transactions, as well as for the deployment and interaction with smart contracts, thus offering an indispensable resource for our simulation needs. Ethers.js was selected for its extensive documentation, ease of use, and its promise of security, having been audited by Ethereum community members.

A dedicated simulation tool plays a pivotal role in our study. This tool is ingeniously devised to mimic the transactional activities of an archetypal user within the Ethereum network and across a selection of L2 solutions. The intent behind this simulation is multifaceted: it aims to methodically quantify and compare the transaction costs, evaluate the speed of transactions, and holistically assess the user experience during various transaction types, including token swaps, ETH transfers, and smart contract interactions.

The tool will be tasked with conducting a series of operations that are representative of a user's interaction with the blockchain. This includes the initiation of transactions such as sending ETH from one address to another, engaging in token exchanges through DEXs (Decentralized EXchanges), such as Uniswap⁵⁰ and SyncSwap⁵¹, and interacting with other smart contracts – in our case bridges⁵². As ETH is the primary currency of the Ethereum network, the simulation will track its flow meticulously, following it as it moves through the

⁵⁰ <https://uniswap.org/>

⁵¹ <https://syncswap.xyz/>

⁵² <https://ethereum.org/developers/docs/bridges>

main Ethereum blockchain onto the L2 solutions and then back again. This journey will illuminate the intricacies of each step in the process, offering us a granular look at the practical implications for users, and enabling a nuanced comparison of the various L2 solutions in operation.

The study also capitalizes on the analytical capabilities of Dune Analytics, a robust platform that allows for the creation of customized dashboards pulling from extensive datasets. With Dune Analytics, we are able to construct a bespoke analytical lens through which we can view and interpret large swathes of blockchain data. The dashboards will be tailored to track economic indicators over the past year, including, but not limited to, transaction fee revenues for the network and specific costs associated with recording data on Layer 1 for the rollups being analyzed. By visualizing these economic indicators, we will be able to discern patterns and trends that are otherwise obfuscated within raw data.

In recognizing the limitations of publicly available data, the study has integrated L2BEAT's API to extend our dataset beyond what is available on their standard website. The API access will allow us to delve deeper into the performance metrics of L2 solutions, capturing data points such as average TPS and the scalability ratios compared to Ethereum. The timeframe for this data extends over the previous year, providing a longitudinal perspective on the evolution and performance of these networks.

This comprehensive approach to data collection and analysis is designed to yield a thorough understanding of the current landscape of Ethereum and its L2 solutions. By capturing a wide spectrum of data, employing a simulation tool for dynamic analysis, utilizing Dune Analytics for economic insights, and integrating L2BEAT's API for extended metrics, this chapter lays a solid foundation for robust analysis and discussion in the subsequent sections of the study. The methodologies outlined herein are critical in ensuring that the research findings are grounded in empirical evidence and offer a credible evaluation of the technologies under scrutiny.

6. Implementation details

6.1 Address generation

The process of address generation is a foundational step in interacting with blockchain networks. For the purpose of this thesis, the generation of Ethereum addresses is crucial to simulate transactions and other user activities. The code segment provided is written in JavaScript and utilizes the `ethers.js` library, a comprehensive collection of functionalities for Ethereum and other EVM-compatible networks.

In the `generateWallet` function, we invoke `ethers.Wallet.createRandom()` which internally uses cryptographic functions to generate a new wallet. This wallet includes a private key and its corresponding public key, from which the Ethereum address is derived. Since Ethereum and EVM networks like Base, zkSync Era, and Arbitrum all share the same address format, this method ensures that the generated address can operate across these networks, given that they all interpret the same cryptographic standards for address formation.

The significance of using the same address across different networks lies in the deterministic nature of wallet generation from a seed phrase. While not explicitly mentioned in the code snippet, if a wallet were to be generated from a specific seed phrase, the resulting address and private key pair would remain consistent across all EVM-compatible chains. This deterministic generation allows for a seamless user experience, where one identity and set of credentials grant access to multiple networks.

In the main function, two separate wallets are generated, each representing a unique user in the simulation. The addresses and private keys are printed to the console (Figure 30.), providing the details necessary for these „users” to interact with the blockchain.

```
hubert@Huberts-Air-2 engineering_thesis % node generateWallets.js
Wallet 1:
Address: 0x3d3cF741A2d0a1Bc6DB979240Fb9a1d5426e2C6b
Private Key: 0x550c20d1c509666f995c28ad92b3393bfdaaa6890f32a9bb25b6af59599b5d9c

Wallet 2:
Address: 0x931a4B8Ab8494364F044E89909B617300545cA3b
Private Key: 0x0d7eac1c579735d2a9a556db33e3b8d5989df44d98b9fd9cfdca0cd0e5facd00
hubert@Huberts-Air-2 engineering_thesis %
```

Figure 30. Example execution of the `generateWallet.js` script, responsible for generating addresses for the further part of the simulation

Source: Author's own work

By designing the simulation in this manner, the thesis can ensure that the data gathered from subsequent steps - such as transacting between addresses, performing token swaps, and bridging between layers - is both accurate and reproducible.

6.2 Sending multiple transactions between two addresses

This section outlines the method and the code used for simulating multiple transactions between two addresses on the Ethereum Sepolia testnet. This example serves as a basis for the

same process that would be conducted on other networks by simply altering the RPC (Remote Procedure Call⁵³) endpoint to the one corresponding to the desired network.

In the provided script, we establish a connection to the Ethereum Sepolia testnet using *ethers.providers.JsonRpcProvider*, specifying the network's RPC URL (Code listing 1.). The script logs a confirmation of this connection, indicating that subsequent transactions will be broadcast on this testnet. To replicate this process on other networks, such as Ethereum Mainnet or an L2 network like Arbitrum, one would only need to replace the RPC URL with that of the target network's node.

```
1. const provider = new ethers.providers.JsonRpcProvider(  
2. "https://rpc.sepolia.org"  
3. );
```

Code listing 1. Connecting to RPC endpoint

In the practical application of the transaction simulation script, the two wallets are instantiated using the *ethers.Wallet* constructor, each initialized with a private key (Code listing 2). In a secure production environment, these private keys would typically be stored and retrieved from an environment file⁵⁴ (.env) to ensure their safety, preventing unauthorized access and exposure in the codebase. However, given that this simulation operates on the Sepolia testnet, the usual security measure of storing these keys in an environment file has been omitted for brevity and simplicity in this illustrative example

```
1. const privateKey1 =  
2. "0xc18d1f9bb42620698593db63b79ec2c481247bc87fc4ad5a0fd3ec5891604116";  
3. const privateKey2 =  
4. "0xd173e761f00885d7acd11789ec93acd72603da5978e0643fe4eeba2ebd80b6d9";  
5.  
6. const wallet1 = new ethers.Wallet(privateKey1, provider);  
7. const wallet2 = new ethers.Wallet(privateKey2, provider);
```

Code listing 2. Initializing wallets

The *sendTransaction* function is an asynchronous operation crafted to handle the sending of Ether from one wallet to another (Code listing 3). It logs the intention to send a transaction, constructs the transaction object with the recipient's address and amount to send, and awaits the send operation. The amount of Ether sent is hardcoded as a small amount suitable for a testnet and would need adjustment depending on the actual network's gas economics. We use *parseEther* to avoid using gwei⁵⁵, which could result in losing a lot of funds in case of any mistake while executing those scripts on mainnets. After sending, the function waits for the transaction to be confirmed on the network, logs the transaction hash, and fetches detailed information like the gas used and the gas price. These details are then appended to a *transactionLog.txt* file, providing a persistent log of all transactions for later analysis.

⁵³ <https://www.alchemy.com/overviews/rpc-node>

⁵⁴ <https://netbasal.com/node-js-v20-6-0-introducing-built-in-env-file-support-fb1b1de4f1fa>

⁵⁵ <https://www.investopedia.com/terms/g/gwei-ethereum.asp>

```

1. async function sendTransaction(senderWallet, recipientAddress) {
2.   const tx = {
3.     to: recipientAddress,
4.     value: ethers.utils.parseEther("0.001"),
5.   };
6.
7.   try {
8.     const transaction = await senderWallet.sendTransaction(tx);
9.     await transaction.wait();

```

Code listing 3. Sending basic transaction to other address

The script then enters a loop to perform five rounds of transactions, alternating between sending from *wallet1* to *wallet2* and vice versa. The loop's iteration count represents the number of transaction pairs; this can be adjusted as needed for more extensive testing. The console logs keep the user informed of the script's progress through each transaction round.

When transactions are being executed, the script logs a completion messages (Figure 31.). This simulation provides insights into the transaction latency and cost on the Sepolia testnet, serving as a proxy for user activity on a live network. Similar transaction patterns will be replicated across different networks, providing comparative data on cost, speed, and efficiency for transactions on Ethereum and chosen Layer 2 solutions.

```

hubert@Huberts-Air-2 engineering_thesis % node sendMultipleTx.js
Starting script...
Connected to Sepolia testnet
Wallets initialized
Transaction round 1/5
Preparing to send transaction from 0xDbDb347B8CBf0CABA8fd775F01978F083577644 to 0xfd186594e9BCA376799901c2EfEf3E0f4E1ca075
Sending transaction: {"to":"0xfd186594e9BCA376799901c2EfEf3E0f4E1ca075","value":{"type":"BigNumber","hex":"0x038d7ea4c68000"}}
Transaction sent, waiting for confirmation...
Transaction confirmed: 0xf94eacfa0f06dd5879c675fc89d1c948d86a0e1f3ba131574a2ca1a5eae495cd
Fetching transaction details for 0xf94eacfa0f06dd5879c675fc89d1c948d86a0e1f3ba131574a2ca1a5eae495cd
Details fetched for transaction: 0xf94eacfa0f06dd5879c675fc89d1c948d86a0e1f3ba131574a2ca1a5eae495cd
Transaction 0xf94eacfa0f06dd5879c675fc89d1c948d86a0e1f3ba131574a2ca1a5eae495cd successful, logged to file
Preparing to send transaction from 0xfd186594e9BCA376799901c2EfEf3E0f4E1ca075 to 0xDbDb347B8CBf0CABA8fd775F01978F083577644
Sending transaction: {"to":"0xDbDb347B8CBf0CABA8fd775F01978F083577644","value":{"type":"BigNumber","hex":"0x038d7ea4c68000"}}
Transaction sent, waiting for confirmation...

```

Figure 31. Example execution of the sendMultipleTx.js script, responsible for sending multiple transactions between two addresses

Source: Author's own work

6.3 Swapping tokens on Uniswap

The section on token swapping will encompass the steps and code necessary for exchanging one cryptocurrency for another within the environment of a decentralized exchange. In this thesis, the transactions will be performed using Uniswap for Ethereum, Base, and Arbitrum, while SyncSwap will be utilized for zkSync Era, due to its specific ecosystem requirements. The code outlined in this section is designed specifically for the Uniswap version only.

Once the connection via RPC is established, the code sets up a wallet instance with a private key and defines the Uniswap smart contract address and ABI (Application Binary Interface⁵⁶), loaded from a local file (Code listing 4.).

⁵⁶ <https://docs.soliditylang.org/en/latest/abi-spec.html>

```

17. const routerAddress = "0xe592427a0aece92de3edee1f18e0157c05861564";
18. const routerABI = JSON.parse(
19.   fs.readFileSync("../jsons/uniswapRouterABI.json", "utf8")
20. );

```

Code listing 4. Initializing details of smart contract to use

The *swapEthForTokens* function is designed to swap Ether for another token (in this case, the UNI token), detailing the transaction parameters such as input amount, recipient, deadline, and the transaction's gas limit (Code listing 5.). It utilizes Uniswap's *exactInputSingle* method, tailored for single-path swaps, and logs the transaction hash upon completion.

```

1. async function swapEthForTokens(tokenOutAddress, amountInEth) {
2.   const tx = await uniswapRouter.exactInputSingle(
3.     {
4.       tokenIn: ethers.constants.AddressZero,
5.       tokenOut: tokenOutAddress,
6.       fee: 3000,
7.       recipient: wallet.address,
8.       deadline: Math.floor(Date.now() / 1000) + 60 * 20,
9.       amountIn: ethers.utils.parseEther(amountInEth),
10.      amountOutMinimum: 0,
11.      sqrtPriceLimitX96: 0,
12.    },
13.    {
14.      value: ethers.utils.parseEther(amountInEth),
15.      gasLimit: gasLimit,
16.    }
17.  );
18.  await tx.wait();
19. }

```

Code listing 5. Initializing *swapEthForTokens* function

The *swapTokens* function follows a similar structure, allowing for the exchange of one token for another, such as UNI⁵⁷ for WETH⁵⁸. Before this transaction can take place, the UNI token must be approved for spending by the Uniswap Router, which is handled by the *approve* method of the token's interface, and the maximum *uint256* is used for the approval amount to avoid needing to approve each transaction (Code listing 6.).

```

1. const approvalAmount = ethers.constants.MaxUint256;
2. const approveTx = await uniToken.approve(routerAddress, approvalAmount);
3. await approveTx.wait();

```

Code listing 6. Approving spending of UNI Token

The main function orchestrates the swap process, starting with ETH to UNI, followed by UNI to WETH, with console logs indicating each step's initiation and completion. Each transaction's result, including any errors, is output to the console and text file, providing real-time feedback on the swap process (Figure 32.).

This token swapping simulation is not only a critical component for understanding the user experience on one network but also serves as a comparative framework to assess the cost, time, and efficiency of similar transactions across different L2 solutions.

⁵⁷ <https://blog.uniswap.org/uni>

⁵⁸ <https://academy.binance.com/en/articles/what-is-wrapped-ether-weth-and-how-to-wrap-it>

```

hubert@Huberts-Air-2 swaps % node swapOnUniswapEthereum
Starting script...
Connected to Sepolia testnet
Uniswap Router ABI loaded
Token addresses set
Contracts initialized
Initiating ETH for UNI swap...
Swapping 0.01 ETH for tokens...
Swapped 0.01 ETH for tokens. Tx Hash: 0x702d02bce278b104b4d13f6370fc4a50d3354c1b788e609c8e37690071f622ec
Approving Uniswap Router to spend UNI...
Uniswap Router approved to spend UNI
Initiating UNI for WETH swap...
Swapping tokens 0x1f9840a85d5aF5bf1D1762F925BDADdC4201F984 for 0xfF9976782d46CC05630D1f6eBAb18b2324d6B14...
Swapped tokens. Tx Hash: 0xd20870d6555834376b05d5537cb76248f5eee1df6618c413a4043aebf1c89e95
Swap completed successfully

```

Figure 32. Example execution of the script responsible for swapping tokens

Source: Author's own work

6.4 Transitioning from L1 to L2 via native bridge

Bridging assets from Ethereum to Layer 2 solutions such as Arbitrum One is a critical function for users looking to interact with faster and more cost-effective networks. This section describes the implementation of using the native bridge to deposit Ether from the Ethereum Sepolia to Arbitrum One Sepolia.

For the bridging operation, the Arbitrum bridge contract's address and ABI on the Sepolia network are defined. The ABI includes a payable *depositEth* function, which is the entry point for users to send Ether to Arbitrum. The bridge contract is instantiated with the wallet signer, allowing transactions to be signed and sent from the wallet's address.

The *depositEthToArbitrum* function is crafted to handle the deposit operation. It logs the action to the console, sends a transaction to the *depositEth* function of the bridge contract with the specified amount of Ether, and waits for the transaction to be confirmed (Code listing 7.). Successful transactions and their details, such as the transaction hash, are logged, providing traceability and transparency to the operation.

```

1. const bridgeContract = new ethers.Contract(
2.   bridgeContractAddress,
3.   bridgeABI,
4.   wallet
5. );
6.
7. async function depositEthToArbitrum(amount) {
8.
9.   const tx = await bridgeContract.depositEth({
10.     value: ethers.utils.parseEther(amount),
11.   });
12.   await tx.wait();

```

Code listing 7. Initializing deposit function

Finally, the main function orchestrates the deposit process. It calls the *depositEthToArbitrum* function with a hardcoded amount, in this case, 0.01 ETH, suitable for demonstration purposes on a testnet. Should an error occur during the process, it is caught and logged to the console, offering an opportunity for debugging and correction.

This implementation provides a practical example of how users can move assets to L2 using native bridges, and similar scripts would be written for other networks like Base and zkSync Era, adapting the contract details and RPC endpoints accordingly. It demonstrates the

necessary steps and considerations for interfacing with L2 networks, forming a basis for understanding the bridging mechanisms integral to L2 functionality and user experience.

6.5 Returning from L2 to L1 via native bridge

Withdrawing assets through native bridges often varies significantly between different networks, each with its own set of rules and verification processes. For instance, withdrawing Ether from Arbitrum One requires a user to initiate a withdrawal transaction and then wait for a seven-day challenge period before finalizing it with an additional transaction. Such nuances are important to understand when interacting with various Layer 2 solutions.

In this section, we will detail the process of withdrawing assets using the zkSync Era bridge as an example. As in all other scripts, a wallet instance is created with a private key, and the bridge contract's address and ABI are defined - this contract address is specific to zkSync's bridging operations.

The withdraw function in the bridge contract is used to initiate a withdrawal from zkSync back to L1. The script logs the intent to withdraw, specifying the recipient address and the amount of Ether to be withdrawn. To accurately account for network demand and ensure transaction completion, the script retrieves the current gas price from the network before sending the transaction (Code listing 8.).

```
1. async function withdraw(address, amount, gasLimit) {
2.
3.   const gasPrice = await provider.getGasPrice();
4.
5.   const tx = await contract.withdraw(address, {
6.     gasLimit: gasLimit,
7.     gasPrice: gasPrice,
8.   });
```

Code listing 8. Initializing withdraw function

The transaction is then sent, and the script actively waits for it to be mined, logging the transaction hash for immediate reference. Once the transaction is complete, the receipt is obtained, providing detailed information such as the amount of gas used and the block number (Figure 33.). These details are crucial as they provide a full audit trail of the withdrawal process and allow for the tracking of the transaction's inclusion in the blockchain.

```
hubert@Huberts-Air-2 withdraws % node withdrawFromzkSyncToEthereum.js
Withdrawing 0.00001 ETH to 0xdbdb347b8cbfb0caba8fd775f01978f083577644...
Waiting for transaction. Transaction Hash: 0x8895c4b18f1b1f5c008be98c4a8fcffc89c8dbb33312aefa2e9237a548770fff
Withdrawal complete. Transaction Hash: 0x8895c4b18f1b1f5c008be98c4a8fcffc89c8dbb33312aefa2e9237a548770fff
Transaction Receipt: {
  to: '0x0000000000000000000000000000000000000000000000000000000000000000',
  from: '0xdbdb347b8cbfb0caba8fd775f01978f083577644',
  contractAddress: null,
  transactionIndex: 0,
  root: '0x040b63062515911d4017946acd16a7b0604bc08dff8a00e05e3b95a71aff11d1',
```

Figure 33. Example execution of the script responsible for withdrawing ETH from L2

The main function handles the execution of the withdrawal, specifying the destination address, the amount to be withdrawn, and the gas limit for the transaction. The specified values, such as the gas limit, may require adjustments based on the network's conditions at the time of the withdrawal (Code listing 9.).

```

45. async function main() {
46.   try {
47.     const destination = "0xdbdb347b8cbfb0caba8fd775f01978f083577644";
48.     const amount = ethers.utils.parseUnits("0.00001", "ether");
49.     const gasLimit = 2897883;
50.     await withdraw(destination, amount, gasLimit);
51.   } catch (error) {
52.     console.error("Error:", error);
53.   }
54. }

```

Code listing 9. Withdrawing ETH from L2 using withdraw function

This implementation showcases the technical considerations and steps needed to perform a withdrawal from a Layer 2 solution back to L1. It underlines the importance of understanding the unique requirements and processes of each network's bridge to successfully navigate the withdrawal phase. Such knowledge is critical when building tools and interfaces that interact with multiple blockchain networks, as it ensures that transactions are not only successful but also optimally executed in terms of cost and time.

6.6 Retrieving data from the L2BEAT API

Although L2BEAT offers valuable insights, it may not always furnish every specific data requirement. In such instances, utilizing their API to independently extract and compute the necessary data becomes essential. This section of the thesis is dedicated to a Python script crafted for retrieving and evaluating data from the API of this platform.

The script employs the *requests* library for executing HTTP requests and the *json* library for decoding JSON (JavaScript Object Notation) responses. The script defines a constant *API_URL* pointing to the L2BEAT API's activity endpoint. A *GET* request is made to this URL, and the response is converted from JSON format into a Python dictionary for ease of data manipulation (Code listing 10.).

```

1. import requests
2. import json
3.
4. API_URL = 'https://api.l2beat.com/api/activity'
5.
6. res = requests.get(API_URL)
7. response = json.loads(res.text)

```

Code listing 10. Connecting to L2BEAT's API endpoint

The script then sets a variable *days* to 365, representing a year's worth of data. It proceeds to extract transaction data for three specific Layer 2 projects: Arbitrum, Base, and zkSync Era. For each project, the script retrieves the last 365 days of daily transaction data. It calculates the total transaction count and the average TPS over this period (Code listing 11.).

```

9. days = 365
10. project1 = 'arbitrum'
11. project2 = 'base'
12. project3 = 'zksync2'
13.
14. # For Arbitrum
15. project_data=response['projects'][project1]['daily']['data'][-days:]
16. project_sum = 0
17. for i in project_data:
18.     project_sum += i[1]
19. print('Arbitrum' + ' ' + str(days) + 'D TX count: ', round(project_sum))
20. print('Arbitrum' + ' average 365D TPS: ', round(project_sum/days)/86400)
21.
22. print("\n")

```

Code listing 11. Calculating transaction data for Arbitrum One

Next, the script gathers combined transaction data for all Layer 2 solutions and for Ethereum separately, over the same period. Again, it computes the total transaction count and average TPS for both these categories (Code listing 12.).

```

44. combined=response['combined']['daily']['data'][-days:]
45. combined_sum = 0
46. ethereum_sum = 0
47. for I in combined:
48.     combined_sum += i[1]
49.     ethereum_sum += i[2]

```

Code listing 12. Calculating transaction data for Ethereum and all L2s

Finally, the script calculates the scaling factor, defined as the ratio of the combined transaction count of all Layer 2 solutions and Ethereum to the transaction count of Ethereum alone (Code listing 13.). This scaling factor serves as a metric to assess the impact of Layer 2 solutions on Ethereum's transaction throughput. The data is then printed in console and used to generate charts using the *matplotlib* library. This chapter provides a comprehensive analysis of transaction data in the context of Ethereum and its Layer 2 scaling solutions, demonstrating the practical application of data extraction and analysis using Python and external APIs.

```

1. scaling_factor = ((combined_sum + ethereum_sum)/ethereum_sum)

```

Code listing 13. Calculating scaling factor

Additionally, this Python script was developed and executed in Google Colab⁵⁹, leveraging the robust computational resources offered by this cloud-based service. This environment not only facilitates efficient running of the script but also simplifies its modularization. Furthermore, Google Colab's flexible platform is conducive to future expansions and enhancements of the script, allowing for seamless integration of additional functionalities or updates as required.

6.7 Retrieving data from the Dune Analytics

Dune Analytics allows to retrieve data by many means, but for the purpose of this thesis SQL language was chosen due to its robustness in managing complex data requests and its proficiency in aggregating and manipulating extensive datasets. The use of SQL is fundamental

⁵⁹ <https://colab.google/>

in my research because of its precision and adaptability in data analysis, which is particularly pivotal when examining the intricate aspects of rollup economics.

The SQL code is organized using Common Table Expressions (CTEs) to create clear and modular subsections of the query. The initial CTE, *batch_fees*, computes the calldata expenses in Ethereum and USD and measures the volume of data for selected projects on a monthly basis (Code listing 14.). Subsequently, *verify_fees* captures monthly data related to the size of proofs and the costs associated with Layer 1 verification. The final CTE, *rev*, consolidates the revenue information for Layer 2.

```

1. WITH batch_fees AS (
2.     SELECT
3.         DATE_TRUNC('month', block_time) AS month,
4.         name AS project_name,
5.         SUM(gas_spent) AS calldata_cost_eth,
6.         SUM(gas_spent_usd) AS calldata_cost_usd,
7.         SUM(data_length) AS data_length
8.     FROM rollup_economics_ethereum.l1_data_fees
9.     WHERE
10.        name IN ('arbitrum', 'zksync era', 'base')
11.    GROUP BY
12.        1,
13.        2
14. ),

```

Code listing 14. Pulling the data for batch fees from Dune Analytics database

The concluding *SELECT* query synthesizes the data from these subqueries, providing an exhaustive monthly economic snapshot for the projects in focus. This includes detailed figures on transactional costs, Layer 2 revenues, and Layer 1 verification expenses. The computation of profits, in Ethereum and USD, is derived by deducting the calculated costs from Layer 2 revenues. Utilizing functions such as *COALESCE* and *CASE* within the query ensures that the absence of data does not impede the results, as it provides the means to handle null values effectively and still deliver precise profit metrics (Code listing 15.).

```

1. data_length / TRY_CAST(1024 AS DOUBLE) / TRY_CAST(1024 AS DOUBLE) AS calldata_mb,
2. COALESCE(r.l2_rev, 0) AS l2_rev,
3. COALESCE(r.l2_rev_usd, 0) AS l2_rev_usd,
4. COALESCE(v.l1_verify_cost_eth, 0) AS l1_verify_cost_eth,
5. COALESCE(v.l1_verify_cost_usd, 0) AS l1_verify_cost_usd,
6. COALESCE(v.size_proof, 0) AS size_proof,
7. CASE
8.     WHEN NOT r.l2_rev IS NULL
9.     THEN r.l2_rev - c.calldata_cost_eth - COALESCE(v.l1_verify_cost_eth, 0)
10.    ELSE 0
11. END AS profit_eth,

```

Code listing 15. Performing data cleansing and disambiguation procedures

The decision to harness Dune Analytics for this component of my research stems from the platform's expansive data accessibility and its simplicity in downloading and graphically representing data with minimal effort. The chapter aims to convey not just the code's methodology, but also its practical significance to the analytical portion of my research in blockchain economics.

7. Results

7.1 Economic analysis of rollups

In the preceding discussions, it has been established that all rollup technologies engage in the practice of posting transaction data on Ethereum's Layer 1. Given the vast volumes of data involved, typically measured in considerable megabytes, this activity stands as their principal cost factor. ZK rollups, such as the zkSync Era, incur additional expenses due to the necessity of validating their proofs on Layer 1. These validations, which are essential to confirm the state and transaction data, represent a notable financial outlay, albeit smaller than the costs associated with posting transaction data.

Delving into the data on incurred costs (Figures 34., 35., 36. and 37.), an average monthly operational cost on Layer 1 for the rollups evaluated is discernible:

- 2001.1 ETH / 3,850,500 USD for Arbitrum One
- 568.6 ETH / 976,700 USD for Base
- 1932.7 ETH / 3,639,250 USD for zkSync Era

These costs are substantial; however, they are expected to decrease significantly with the implementation of anticipated updates such as the EIP-4844 upgrade. Additionally, the teams behind these projects are actively working on innovative solutions to reduce the quantity of data required to reconstruct their respective states.

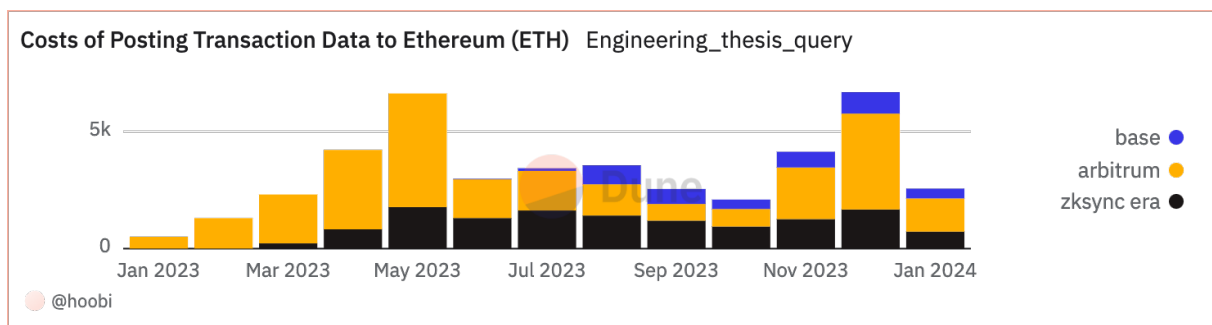


Figure 34. Chart of costs of posting transaction data to Ethereum for chosen L2s (ETH)
Source: Author's own work using Dune Analytics

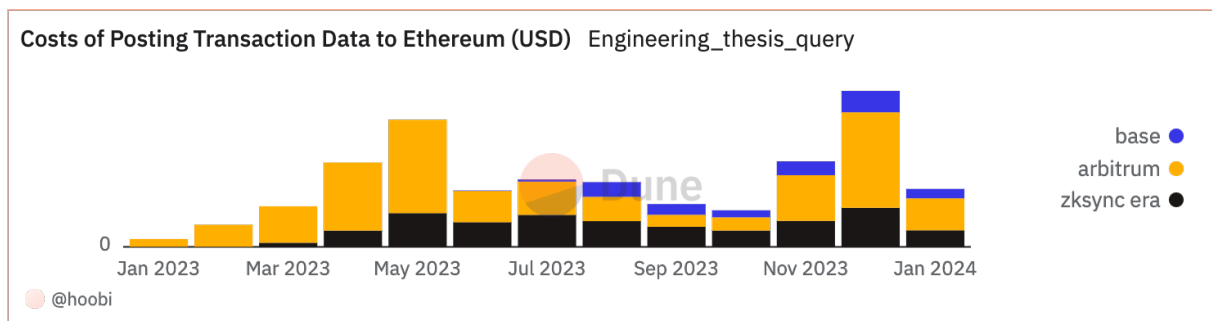


Figure 35. Chart of costs of posting transaction data to Ethereum for chosen L2s (USD)
Source: Author's own work using Dune Analytics

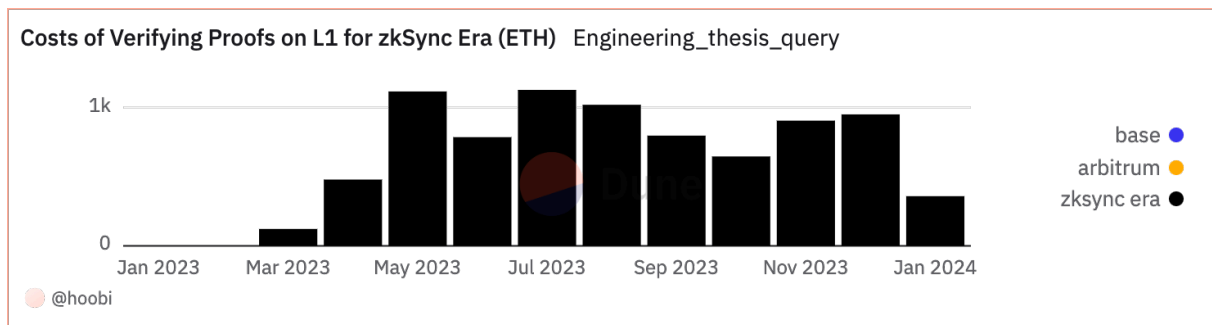


Figure 36. Chart of costs of verifying proofs on Ethereum for zkSync Era (ETH)

Source: Author's own work using Dune Analytics



Figure 37. Chart of costs of verifying proofs on Ethereum for zkSync Era (USD)

Source: Author's own work using Dune Analytics

Looking at the revenue side, rollups benefit from substantial inflows of ETH as they employ this cryptocurrency as their native gas token. Referring to the data (Figures 38. and 39.), the average monthly revenue from transaction fees for the rollups is:

- 2744.1 ETH / 5,280,175 USD for Arbitrum One
- 1069.7 ETH / 1,837,453 USD for Base
- 3061.5 ETH / 5,764,766 USD for zkSync Era

While project teams are committed to reducing fees with each upgrade, the user base and the volume of prominent projects built on these L2 infrastructures, such as Worldcoin, are expected to expand. Consequently, the number of transactions per user is likely to increase, ensuring a rise in revenue streams.

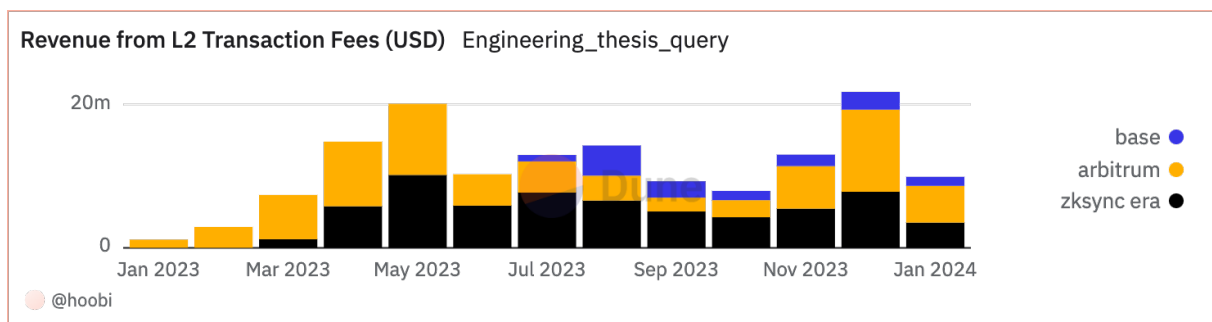


Figure 38. Chart of revenue from L2 transaction fees (USD)

Source: Author's own work using Dune Analytics

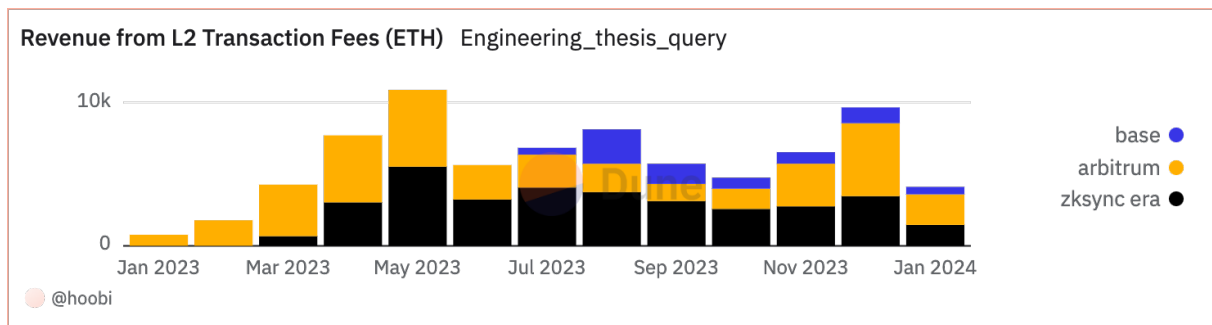


Figure 39. Chart of revenue from L2 transaction fees (ETH)
Source: Author's own work using Dune Analytics

When these two financial metrics - costs and revenues - are juxtaposed, we can derive the on-chain profitability for each of these projects (Figures 40. and 41.):

- 743 ETH / 1,429,675 USD for Arbitrum One
- 501.1 ETH / 860,653 USD for Base
- 1128.8 ETH / 2,125,516 USD for zkSync Era

The data indicates that these rollup solutions occupy a profitable niche within the blockchain ecosystem, and their profitability is poised to increase as their respective ecosystems evolve. Notably, among the three projects listed, only Arbitrum has distributed its own token, ARB, while Base and zkSync have not, which may influence profitability.

However, it is essential to consider the potential inflation of transaction volumes on networks like zkSync Era and Base due to users attempting to qualify for potential airdrops⁶⁰ of those L2 tokens. A subsequent analysis, conducted several months after such airdrops, would provide further insights into the genuine user engagement and profitability post-event.

Additionally, the data reveals that a rollup's profitability is not necessarily correlated with its TVL; rather, zkSync Era demonstrates a pronounced edge over Arbitrum in this respect.

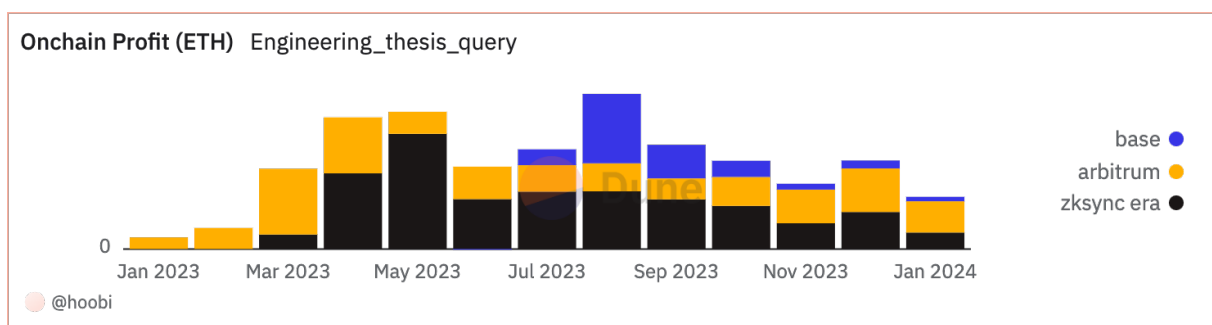


Figure 40. Chart of onchain profit of chosen L2 solutions (ETH)
Source: Author's own work using Dune Analytics

⁶⁰ <https://globalcoinresearch.com/2023/06/16/the-effectiveness-of-l2-airdrop/>

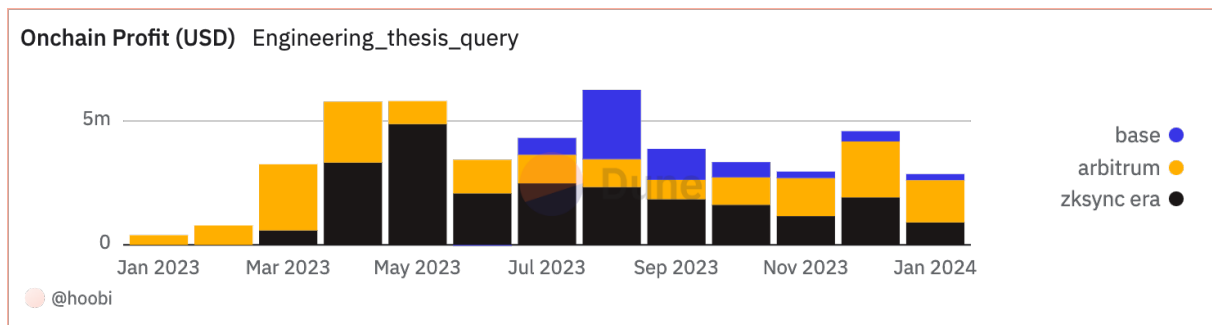


Figure 41. Chart of onchain profit of chosen L2 solutions (USD)

Source: Author's own work using Dune Analytics

7.2 Analysis of activity on chosen networks

Drawing from the data collated from L2BEAT, the preceding analysis has underscored the transactional efficiency of Layer 2 solutions when juxtaposed with Ethereum's Layer 1. The data culled over a year manifests that platforms such as Arbitrum, Base, and zkSync Era, in isolation, have augmented Ethereum's transaction handling capacity by a margin nearing 50% (Figure 42.). This increment not only illustrates the efficacy of L2 solutions in bolstering Ethereum's performance but also their pivotal role in addressing the network's congestion.

In a broader scope, encompassing the collective strength of the L2 ecosystem, an average monthly scaling factor emerges at 4.31. This number delineates the multiplicative effect L2 solutions have on Ethereum's transactional throughput. The implications are considerable, suggesting that the L2 ecosystem could potentially quadruple Ethereum's capacity for processing transactions monthly, a leap forward in scalability that could address the burgeoning demands of a growing user base and the complexity of emerging decentralized applications.

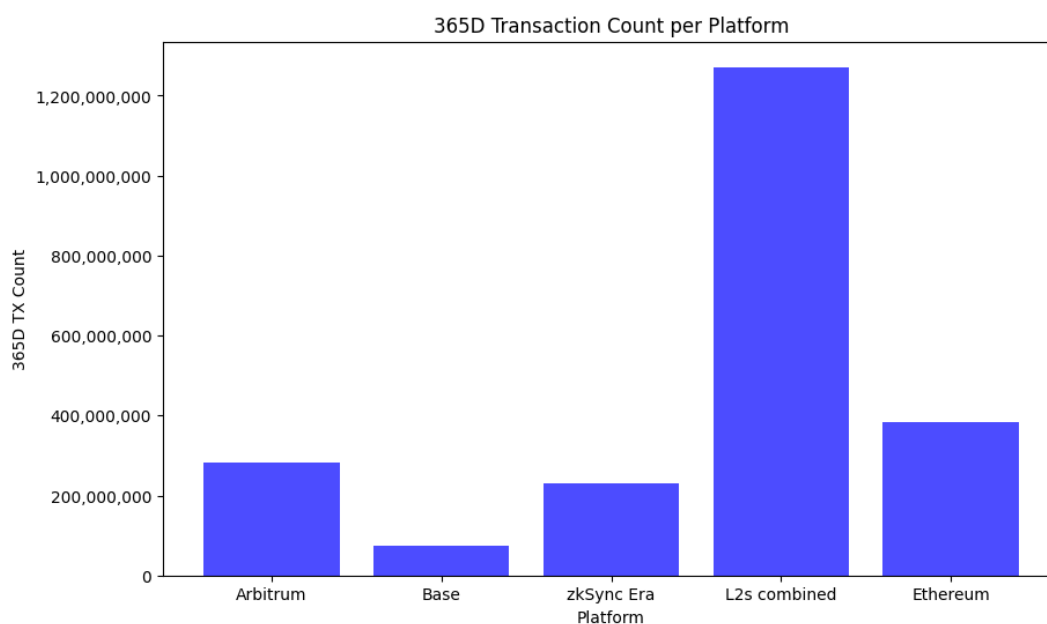


Figure 42. Chart of the total number of transactions performed on evaluated networks

Source: Author's own work

The examination of TPS (Figure 43.) further cements the strategic importance of L2 solutions. The data indicates that all L2 solutions combined markedly surpasses Ethereum in TPS, delineating a high-performance profile that is crucial for transaction processing efficiency. Higher TPS correlates directly to an enhanced user experience, characterized by rapid transaction validation and minimal latency, which are essential attributes in the context of blockchain utility and adoption.

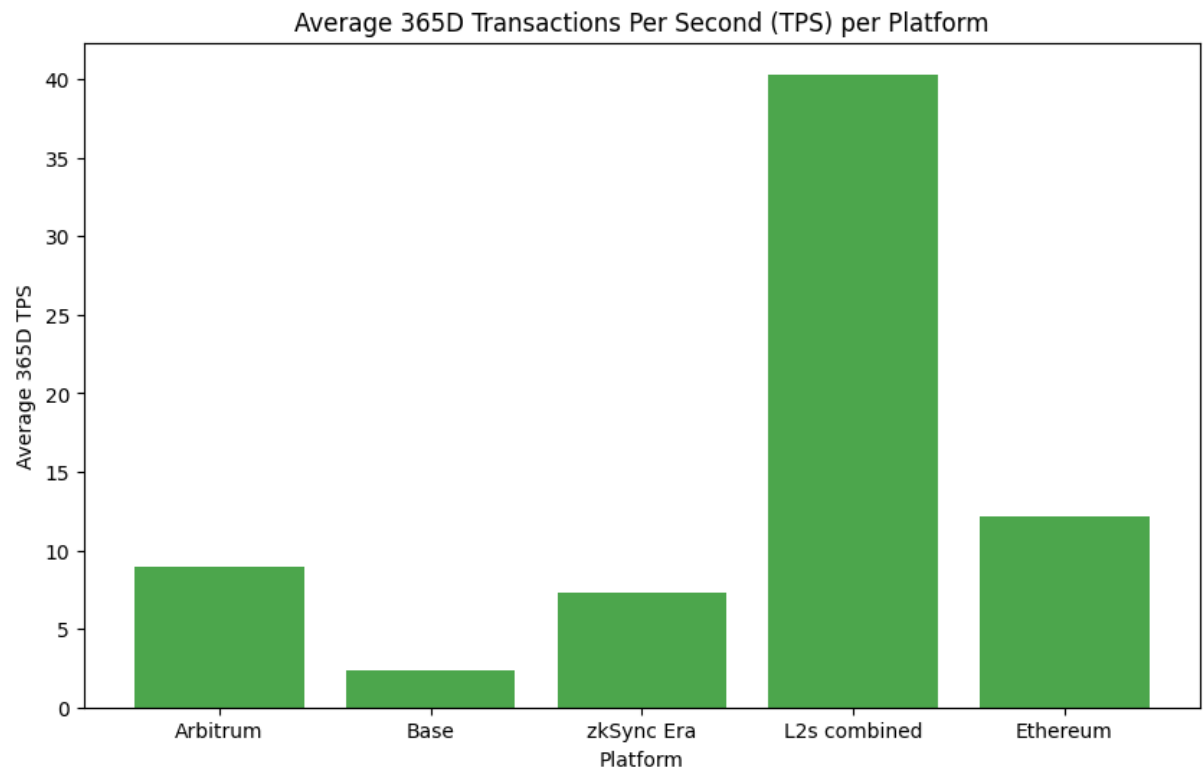


Figure 43. Chart of average monthly TPS for last year for evaluated networks
Source: Author’s own work

Collectively, the data on transaction count and TPS form a compelling narrative of the L2 ecosystem's robustness. They indicate not just an aggregate ability to handle a larger volume of transactions but also a rapid processing capability that stands to refine the blockchain experience significantly.

In light of this paper’s focus, these findings are integral in evaluating the technical and economic landscape of blockchain scalability solutions. They underscore the significance of L2 platforms in the blockchain infrastructure and presage a trajectory towards heightened adoption of these solutions. As such, they paint an optimistic picture of the future, one where L2 adoption is likely to redefine the paradigms of transaction processing and efficiency within the blockchain domain.

7.3 Analysis of user experience on chosen networks

Analysis of the data harvested through our proprietary tool simulating user activity on assessed networks reveals a marked cost-efficiency in executing transactions on Layer 2 solutions as opposed to Ethereum's mainnet (Table 4.). This economic advantage persists even when incorporating the utilization of native bridges for the transfer of assets into and out of the selected Layer 2 platforms.

Table 4. Combined details of transactions executed by the simulation tool on Ethereum
Source: Author's own work based on data gathered by simulation tool

Time needed to complete ten standard transactions	61.1 s
Time needed to complete swaps	11.2 s
Total time needed to complete all transactions	72.3 s
Cost of sending ten standard transactions in USD	24.12 USD
Cost of sending ten standard transactions in ETH	0,0097 ETH
Cost of swapping tokens in USD	25.29 USD
Cost of swapping tokens in ETH	0,010 ETH
Total cost of all transactions in USD	49.41 USD
Total cost off all transactions in ETH	0,020 ETH

The least cost-effective and “slow” network turned out to be Arbitrum (Table 5.). Conducting the transactions of interest on this network could save the user 32.9 USD. On the other end, Base was the cheapest (Table 6.) and offered a saving of over 44.19 USD. It is important to note, however, that Base is significantly younger than Arbitrum and has fewer regular users, which contributes to its lower fees and shorter transaction finalization times.

Table 5. Combined details of transactions executed by the simulation tool on Arbitrum One
Source: Author's own work based on data gathered by simulation tool

Time needed to complete ten standard transactions	46 s
Time needed to complete swaps	9 s
Time needed to deposit funds on L2 via native bridge	11 m 52 s
Time needed to claim back funds on L1 via native bridge	7D
Total time needed to complete all transactions	12 m 47 s + 7D
Cost of sending ten standard transactions in USD	1.32 USD
Cost of sending ten standard transactions in ETH	0,00053 ETH
Cost of using native bridge (both ways combined) in USD	14.44 USD
Cost of using native bridge (both ways combined) in ETH	0,0058 ETH
Cost of swapping tokens in USD	0.75 USD
Cost of swapping tokens in ETH	0,00030 ETH
Total cost of all transactions in USD	16.51 USD
Total cost off all transactions in ETH	0,0066 ETH

Table 6. Combined details of transactions executed by the simulation tool on Base
Source: Author's own work based on data gathered by simulation tool

Time needed to complete ten standard transactions	36 s
Time needed to complete swaps	25 s
Time needed to deposit funds on L2 via native bridge	4 m 45 s
Time needed to claim back funds on L1 via native bridge	7D + 2h
Total time needed to complete all transactions	5m 46 s + 7D + 2h
Cost of sending ten standard transactions in USD	0.9 USD
Cost of sending ten standard transactions in ETH	0,00036 ETH
Cost of using native bridge (both ways combined) in USD	3.65 USD
Cost of using native bridge (both ways combined) in ETH	0,0015 ETH
Cost of swapping tokens in USD	0.67 USD
Cost of swapping tokens in ETH	0,00027 ETH
Total cost of all transactions in USD	5.22 USD
Total cost off all transactions in ETH	0,0021 ETH

Intriguing insights also emerge from examining the zkSync Era (Table 7.). Despite the theoretical expectation for ZK rollups to be cheaper and faster, this is not necessarily the reality. It is true that they significantly outperform in terms of the time it takes to withdraw funds, but this does not compensate for the costlier and slower transactions compared to optimistic rollups. Nonetheless, it is essential to consider the specific activities related to the zkSync token airdrop, which likely contribute to this situation.

Regarding the finality of our transactions, while they tend to be somewhat quicker on L2 networks due to specific implementation characteristics, there are significant latencies involved. Transferring assets via a bridge from L1 to L2 can take several minutes, and there are notable delays when withdrawing them - one week for optimistic rollups and 24 hours for zkSync Era. It's worth noting that the withdrawal time for zkSync Era is set by the development team and is anticipated to be reduced to approximately two hours in the future. Hence, while it is possible to economize on transactions, one must be cognizant of the risks and issues concerning the decentralization of L2 solutions. If there's an intent to revert to L1 immediately after operations on the targeted smart contracts, this could potentially lock funds for an extended duration.

It is important to note that the measurements were conducted during a particularly calm period when the fees on Ethereum were exceptionally low. Nevertheless, the cost-benefit of using L2 is clear and would only increase as traffic on L1 grows, further highlighting the importance, necessity, and advantages of L2 in certain aspects. In our measurements, we also excluded the transaction that allows a DEX smart contract to utilize a chosen token during a secondary swap due to its negligible cost and execution time. The price of Ether was chosen based on the day the test was conducted, that is January 19, 2024.

Table 7. Combined details of transactions executed by the simulation tool on zkSync Era
Source: Author's own work based on data gathered by simulation tool

Time needed to complete ten standard transactions	39 s
Time needed to complete swaps	14 s
Time needed to deposit funds on L2 via native bridge	17 m 49 s
Time needed to claim back funds on L1 via native bridge	24 h
Total time needed to complete all transactions	18 m 42 s + 24 h
Cost of sending ten standard transactions in USD	1.50 USD
Cost of sending ten standard transactions in ETH	0,00060 ETH
Cost of using native bridge (both ways combined) in USD	8.80 USD
Cost of using native bridge (both ways combined) in ETH	0,0035 ETH
Cost of swapping tokens in USD	0.47 USD
Cost of swapping tokens in ETH	0,00019 ETH
Total cost of all transactions in USD	10.77 USD
Total cost off all transactions in ETH	0,0043 ETH

8. Summary and conclusions

In summary, this engineering thesis has been devoted to a thorough analysis of Layer 2 solutions and assessing their capabilities to scale the Ethereum network. The study found that while these solutions are currently profitable, generating revenue, and attracting many users with rapid transaction speeds and low costs, they are still in the nascent stages of development and should be regarded as experimental. However, it is certain that there will be an uptick in their adoption, driven by an increase in the user base of currently deployed solutions as well as the introduction of new ones, and the resolution of existing short-term technical issues like level of decentralization or risks of permanently losing funds is just a matter of time. This progress will be facilitated by forthcoming upgrades to both Ethereum, such as EIP-4844 and the Layer 2 projects themselves. In 2024, we may witness the emergence of the first widely adopted Stage 2 rollup in Arbitrum, as well as many transitions from Stage 0 to Stage 1 rollups like zkSync Era and Base. Using this knowledge, we can conclude that the rollup-centric Ethereum roadmap was a success. It has maintained Ethereum's relevance in the broader blockchain landscape and marked a shift in its approach from client-to-business to business-to-business.

One possibility that comes to mind is the potential stalling of the development or popularity of these solutions due to major controversies, such as the hacking of all funds from a selected Layer 2 network, whether through a flaw in the infrastructure's code or deliberate actions by the creators. As can be observed in the section evaluating upgradeability and the overall architecture of smart contracts – this is entirely possible and probably just a matter of time.

While rollups currently provide a substantial fivefold increase in transaction capacity, the stark reality is that Ethereum must scale by 10000x, or even 100000x, to achieve widespread adoption. Long term viability of these safest Layer 2 solutions is uncertain due to insurmountable challenges. Issues such as liquidity fragmentation, governance complexities, varied liveness attributes, and the inherent risks of smart contracts suggest that Layer 2 solutions could at best be a temporary relief, effective for perhaps only the next decade or two. To truly reach the desired scalability, it may become necessary to implement complex and significant modifications to Ethereum's underlying code.

The analysis presented in this engineering thesis allows everyone to independently evaluate the balance of benefits in relation to the potential risks of using L2 solutions and to make an informed decision about transferring their assets onto these platforms.

This work paves the way for further research that could extend to a broader evaluation of more projects, including a wider array of rollups like Linea, Scroll, or Starknet, as well as representatives of Validium and Optimium technologies. A potential avenue for future development could also involve a focus on Data Availability by analyzing projects such as Celestia, Avail, and Eigen DA and their influence on the evolution of L2 solutions.

In conclusion, this thesis offers a significant contribution to understanding the mechanisms of Layer 2 solutions and their future market presence. It serves as an excellent gateway into the intricate and expansive subject of Ethereum scaling, aptly described as a rabbit hole due to its depth and complexity.

References

- [1] Vitalik Buterin. *A rollup-centric Ethereum roadmap*. [Access: December 11, 2023]. Available online: <https://ethereum-magicians.org/t/a-rollup-centric-ethereum-roadmap/4698>
- [2] Vitalik Buterin. *Ethereum. A Next-Generation Smart Contract and Decentralized Application Platform*. [Access: December 11, 2023]. Available online: https://ethereum.org/669c9e2e2027310b6_b3cdce6e1c52962/Ethereum_Whitepaper_-_Buterin_2014.pdf
- [3] Mirko Bez, Giacomo Fornari, Tullio Vardanega. *The scalability challenge of Ethereum: An initial quantitative analysis*. [Access: December 11, 2023]. Available online: <https://ieeexplore.ieee.org/document/8705874>
- [4] Ionut Rosca, Alexandra Ina-Butnaru, Emil Simion. *Security of Ethereum Layer 2s*. [Access: December 11, 2023]. Available online: <https://eprint.iacr.org/2023/124.pdf>
- [5] Vitalik Buterin. *An Incomplete Guide to Rollups*. [Access: December 11, 2023]. Available online: <https://vitalik.eth.limo/general/2021/01/05/rollup.html>
- [6] Chainlink. *What is Layer 2*. [Access: December 11, 2023]. Available online: <https://chain.link/education-hub/what-is-layer-2>
- [7] Joseph Poon, Vitalik Buterin. *Plasma: Scalable Autonomous Smart Contracts* [Access: December 11, 2023]. Available online: <https://plasma.io/plasma-deprecated.pdf>
- [8] Ethereum Foundation. *Validium*. [Access: December 11, 2023]. Available online: <https://ethereum.org/en/developers/docs/scaling/validium/>
- [9] Avihu Levy, Uri Kolodny. *Validity Proofs vs. Fraud Proofs*. [Access: December 11, 2023]. Available online: <https://medium.com/starkware/validity-proofs-vs-fraud-proofs-4ef8b4d3d87a>
- [10] Shafi Goldwasser, Silvio Micali, Charles Rackoff. *The knowledge complexity of interactive proof systems*. SIAM J. Comput., 1989.
- [11] Ed Felten. *Why I'm still optimistic about optimistic*. [Access: December 11, 2023]. Available online: <https://medium.com/offchainlabs/why-im-still-optimistic-about-optimistic-41376367d4f9>
- [12] Vitalik Buterin. *Proposed milestones for rollups taking off training wheels*. [Access: December 11, 2023]. Available online: <https://ethereum-magicians.org/t/proposed-milestones-for-rollups-taking-off-training-wheels/11571>
- [13] Luca Donno. L2BEAT. *Introducing Stages — a framework to evaluate rollups maturity*. [Access: December 11, 2023]. Available online: <https://medium.com/l2beat/introducing-stages-a-framework-to-evaluate-rollups-maturity-d290bb22befe>
- [14] Adrian Adamiak, Bartek Kiepuszewski, Krzysztof Urbański, Luca Donno, Piotr Szlachciak. *Upgradeability of Ethereum L2s*. [Access: December 11, 2023]. Available online: <https://drive.google.com/file/d/182ycEW8C2wk5tGd3X1tG8oQfUy9WmSJk/view>
- [15] Harry Kalodner, Steven Goldfeder, Xiaoqi Chen, S. Matthew Weinberg, Edward W. Felten. *Arbitrum: Scalable, private smart contracts*. [Access: December 11, 2023].

- Available online: <https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-kalodner.pdf>
- [16] Offchain Labs, Inc. *Arbitrum Nitro: A Second-Generation Optimistic Rollup*. [Access: December 11, 2023]. Available online: [https://github.com/OffchainLabs/nitro/blob/master/docs/Nitro-white paper.pdf](https://github.com/OffchainLabs/nitro/blob/master/docs/Nitro-white%20paper.pdf)
 - [17] Jesse Pollak. *Meet Base*. [Access: December 11, 2023]. Available online: https://base.mirror.xyz/jjQnUq_UNTQOk7psnGBFOsShi7FlrRp8xevQUipG_Gk
 - [18] Matter Labs. *gm zkEVM*. [Access: December 11, 2023]. Available online: <https://blog.matter-labs.io/gm-zkevm-171b12a26b36>
 - [19] Worldcoin Foundation. *Worldcoin tech tree*. [Access: December 11, 2023]. Available online: <https://worldcoin.org/tech-tree#protocol>
 - [20] Vitalik Buterin. *EIP 4844, Proto-danksharding FAQ*. [Access: December 11, 2023]. Available online: https://notes.ethereum.org/@vbuterin/proto_danksharding_faq

Figures

Figure 1. Vitalik Buterin announces rollup-centric Ethereum roadmap	4
Figure 2. Chart of total value in USD of all tokens on L2 networks	5
Figure 3. Josh Stark commenting L2 vs L1 TVL.....	5
Figure 4. Chart of TPS (Transactions Per Second) on Ethereum mainnet and all L2 networks for last year.....	8
Figure 5. The blockchain trilemma visualization.....	9
Figure 6. Part of the discussion about upcoming changes to the L2 landscape by Dankrad Feist and David Hoffman	10
Figure 7. Six scaling solutions with the most market share	12
Figure 8. Visualization of the rollup transaction posting	12
Figure 9. TVL growth of all optimistic rollups combined	14
Figure 10. All currently deployed optimistic rollups and their details.....	14
Figure 11. TVL growth of all ZK rollups combined.....	15
Figure 12. All currently deployed ZK rollups and their details	15
Figure 13. Arbitrum One details and summary	18
Figure 14. Arbitrum One TVL growth.....	18
Figure 15. Arbitrum One activity growth compared to Ethereum	19
Figure 16. Arbitrum One risk analysis visualization.....	20
Figure 17. Upgrade system architecture of Arbitrum One.....	22
Figure 18. Base details and summary.....	23
Figure 19. Base TVL growth.....	23
Figure 20. Base activity growth compared to Ethereum	23
Figure 21. Base risk analysis visualization	24
Figure 22. Upgrade system architecture of Base (Optimism).....	26
Figure 23. zkSync Era details and summary	27
Figure 24. zkSync Era TVL growth	27
Figure 25. zkSync Era activity growth compared to Ethereum	28
Figure 26. zkSync Era risk analysis visualization.....	29
Figure 27. Upgrade system architecture of zkSync Era	31
Figure 28. A photo of an orb, a device used to scan the iris	32
Figure 29. Details about the activity of Worldcoin users.....	33
Figure 30. Example execution of the generateWallet.js script, responsible for generating	37
Figure 31. Example execution of the sendMultipleTx.js script,	39
Figure 32. Example execution of the script responsible for swapping tokens	41
Figure 33. Example execution of the script responsible for withdrawing ETH from L2	42
Figure 34. Chart of costs of posting transaction data to Ethereum for chosen L2s (ETH).....	46
Figure 35. Chart of costs of posting transaction data to Ethereum for chosen L2s (ETH)	46
Figure 36. Chart of costs of verifying proofs on Ethereum for zkSync Era (ETH)	47
Figure 37. Chart of costs of verifying proofs on Ethereum for zkSync Era (USD).....	47
Figure 38. Chart of revenue from L2 transaction fees (USD).....	47
Figure 39. Chart of revenue from L2 transaction fees (ETH).....	48
Figure 40. Chart of onchain profit of chosen L2 solutions (ETH).....	48
Figure 41. Chart of onchain profit of chosen L2 solutions (USD).....	49
Figure 42. Chart of the total number of transactions performed on evaluated networks	49
Figure 43. Chart of average monthly TPS for last year for evaluated networks.....	50

Tables

Table 1. Decentralization assessment of Arbitrum One.....	21
Table 2. Decentralization assessment of Base	25
Table 3. Decentralization assessment of zkSync Era	30
Table 4. Combined details of transactions executed by the simulation tool on Ethereum	51
Table 5. Combined details of transactions executed by the simulation tool on Arbitrum One	51
Table 6. Combined details of transactions executed by the simulation tool on Base	52
Table 7. Combined details of transactions executed by the simulation tool on zkSync Era....	53