SAPIENZA
UNIVERSITÀ DI ROMA

BLOCKCHAIN
AND DISTRIBUTED LEDGER TECHNOLOGIES

OPTIMISTIC ROLLUPS

Arbitrum - Base - Optimism

Submitted by:

Marildo Cani - 2246171

Under the guidance of:

Prf: Massimo La Morgia

ABSTRACT

Ethereum's Layer-2 (L2) rollups extend the security of Ethereum while targeting higher throughput and lower transaction costs. However, their real performance characteristics are often presented from a theoretical, marketing, or architectural perspective, rather than based on reproducible empirical measurement. This report provides a structured and data-driven analysis of three major Optimistic Rollups: Base, Optimism, and Arbitrum, focusing on user-observable behavior rather than assumed theoretical performance.

Recent L2 transaction data was collected directly through public RPC endpoints and processed through a custom measurement pipeline developed for this work. The data was cleaned, normalized into a unified parquet dataset, and analyzed to quantify inter-block times, L2 transaction latency distributions, fee scaling under varying calldata payload sizes, and L2-L1 posting delays. To evaluate security implications, a standardized three-stage Finality Pyramid is introduced (L2-Included, L2-Safe, L1-Final/Withdrawable), together with a Trust and Maturity Matrix covering proof mechanisms, challenge windows, DA paths (calldata vs blobs under EIP-4844), sequencer centralization, and upgrade governance.

The empirical results confirm near-linear fee scaling with payload size across all three rollups, while exposing measurable differences in latency variability and posting behavior that relate to each rollup's batching strategy and data availability model. By combining architectural properties with direct performance measurements, this work offers a grounded view of what guarantees L2 users, developers, and bridging systems can rely on in practice, beyond theoretical claims or ecosystem narratives.

Keywords: Ethereum , Layer-2 , Optimistic Rollups , Base , Optimism , Arbitrum , Transaction Fees , Finality Py ramid , Trust Matrix , EIP-4844 , Data Availability

INDEX

Chapter 1 — Introduction and Motivation..... 6

Chapter 2 — Background..... 9

2.1 Blockchains: transactions, blocks, consensus, finality..... 9

2.2 Ethereum recap: accounts, gas, calldata, base fee, receipts..... 9

2.3 Rollups: why scalability, how they work..... 10

2.4 OP Stack vs Arbitrum Nitro..... 11

2.5 Finality concepts: L2 included, L2 safe, L1 posted, L1 final..... 12

2.6 Data availability: calldata vs blobs..... 12

Chapter 3 — Related Work..... 14

3.1 Architectural guarantees: optimistic rollups in practice..... 14

3.2 Data availability economics and EIP-4844..... 15

3.3 Measurements ,surveys and frameworks..... 15

3.4 Sequencers, MEV, and decentralization..... 15

Chapter 4 — Systems Overview & Governance..... 18

4.1 Launch Dates and Vision..... 18

4.2 Governance and Upgrade Process..... 18

4.3 Economic Model Basics..... 19

4.4 Developer Ecosystem Notes..... 20

Chapter 5 — Methodology..... 21

5.1 Data Sources..... 21

5.2 Data Collection..... 22

5.3 Cleaning and Enrichment..... 22
5.4 Metrics..... 23
5.5 Visuals and KPIs..... 24
5.6 Reproducibility..... 25

Chapter 6 — Datasets..... 26
6.1 Collection Windows..... 26
6.2 Row Counts, Deduplication, and Missing Fields..... 26
6.3 Ethical and Data Quality Considerations..... 27

Chapter 7 — Empirical Results and Analysis..... 29
7.1 Network Activity Composition..... 29
7.2 Latency B — Inter-Block Time..... 31
7.3 Latency A — Upper Bound to First Inclusion.....34
7.4 Latency C — L2 to L1 Posting Delay..... 35
7.5 Fees vs Payload — Cost per KiB Scaling (RQ2)..... 37
7.6 Sanity Checks and Sensitivity..... 38

Chapter 8 — Discussion..... 39
8.1 Research Questions — Interpretation of Results..... 39
8.2 Finality Pyramid — Practical Settlement Horizons..... 40
8.3 Trust and Maturity Matrix Interpretation..... 42
8.4 Risks and Assumptions..... 43

Chapter 9 — Security and Risk Profile..... 45
9.1 Known Incidents and Outage Realities..... 45
9.2 Centralization Vectors..... 45

9.3 Data Availability Risk..... 46
9.4 MEV, Censorship, and Sequencer Power..... 47
9.5 Summary of Risk Perspective..... 47

Chapter 10 — Tooling and Developer Experience..... 48
10.1 EVM Compatibility and Development Flow..... 48
10.2 RPC Access and Node Interaction..... 48
10.3 SDKs, Documentation, and Verification Surface..... 49
10.4 DX Perspective Relative to Findings..... 49

Chapter 11 — Comparison Against Ethereum L1..... 50
11.1 Security and Decentralization..... 50
11.2 Performance and Latency Comparison..... 50
11.3 Adoption and Developer Environment..... 51
11.4 Economics and Fee Model Interpretation..... 51
11.5 Positioning Conclusion..... 51

Chapter 12 — Conclusion and Future Work..... 52
Contributions..... 54
REFERENCES AND BIBLIOGRAPHY.....56

Introduction and Motivation

Ethereum is one of the most widely used public smart contract platforms, but its base layer throughput is limited and transaction fees can become prohibitive during periods of network congestion. As adoption expands across DeFi, gaming, payments, social applications, and enterprise use-cases, execution at global scale on Layer 1 becomes economically inefficient and technically constrained. Layer-2 rollups emerged as an architectural response: they aim to increase transaction throughput and significantly reduce cost while maintaining Ethereum as the ultimate security and settlement layer.

Layer-2 rollups execute transactions off-chain on a separate environment, then publish compressed state data back to Ethereum in order to inherit its security guarantees. Among the different rollup designs, Optimistic Rollups have gained rapid adoption because they are easier to deploy today at scale, compatible with existing EVM tooling, and well integrated across major infrastructures and exchanges. This report focuses on three major Optimistic Rollups: Optimism, Base, and Arbitrum. They are today among the dominant scaling solutions used by real users, applications, liquidity, and exchanges, and therefore represent a meaningful subset for evaluating the present maturity and practical performance of Ethereum L2 systems.

Despite their popularity, discussions on L2 performance often remain theoretical or marketing-driven. There is limited empirical work measuring the actual behavior experienced by end users under real network conditions. While official documentation describes the architectural guarantees, it does not directly quantify transaction latency distributions, fee behavior as calldata size changes, or the practical time between L2 inclusion and L1 anchoring. For developers building applications, for researchers studying system properties, and for bridges interacting with multiple chains, understanding these practical properties is essential. A scientific evaluation of real traces and recent blocks is required to

determine how these systems behave in practice rather than how they are expected to behave in idealized design assumptions.

The purpose of this report is to bridge the gap between architectural descriptions and empirical reality. The analysis is based on direct data collection from public RPC endpoints and explorer APIs, combined with a reproducible pipeline which transforms raw transactions into a uniform dataset suitable for comparison. The focus is on measuring and explaining user-visible behavior. This ensures that claims throughout the document are grounded on verifiable evidence rather than assumed theoretical performance.

The work proceeds through three core research questions:

RQ1: What are realistic Layer-2 inter-block times on Base, Optimism, and Arbitrum, observed under normal operational conditions.

RQ2: How do fees scale as transaction calldata size increases, and can this be used as a proxy for cost per KiB across the three L2s.

RQ3: What are the posting delays between L2 inclusion and L1 anchoring, and how does this affect the practical interpretation of finality from a user and protocol perspective.

To address these questions, this report contributes the following elements:

• A reproducible Python-based data pipeline which collects recent L2 transactions, performs dataset normalization, and exports parquet files for longitudinal experimentation.

• A cleaned dataset enhanced with derived metadata such as contract versus externally owned address classification and transaction type buckets.

• Empirical visualizations including latency distributions, cumulative distribution functions, fee versus payload behavior, and transaction composition summaries.

• Implementation of the first execution layer of Latency C measurements, together with instrumentation infrastructure for capturing future batching windows and posting delays.

• A structured synthesis model introduced later in the report, including a Finality Pyramid and a Trust and Maturity Matrix, which consolidate architectural differences with the empirical findings.

This report starts with a foundational background chapter to ensure that the reader does not require prior blockchain experience or domain knowledge. Concepts such as sequencers, data availability, fraud proofs, challenge windows, and the distinction between L2 inclusion and L1 final settlement are explained from first principles. After the background, the report analyses the architecture and governance of the three rollups, reviews relevant academic literature, describes the methodology used to collect and process data, presents experimental findings, and interprets them in relation to system design choices. The document concludes by synthesizing the empirical results into a structured security and trust profile, highlighting current maturity levels and noting open research directions for future L2 evolution.

Background

2.1 Blockchains: transactions ,blocks ,consensus ,finality

A blockchain is a distributed ledger that allows multiple independent participants to agree on a shared state without trusting each other directly. Users submit signed transactions which express state changes (for example: transfer of value, or execution of contract logic).A set of transactions is grouped into a block, and blocks are appended over time forming a chain. Every node verifies that transactions follow the protocol rules before accepting them.

Consensus is the mechanism by which the network agrees which block becomes the next canonical block. Different blockchains adopt different consensus mechanisms (Proof-of-Stake, Proof-of-Work, or variants of BFT-derived systems). Consensus is what prevents double-spending and ensures that all honest nodes converge on the same global view of state [6].

Finality represents the moment at which a transaction is considered irreversible with extremely high probability. Finality can be probabilistic (as in longest chain PoW) or economically guaranteed (as in PoS with slashing). The important idea is that users, dApps, and bridges make security decisions based on the strength and time required for finality [7].

2.2 Ethereum recap: accounts, gas, calldata, base fee, receipts

Ethereum uses an account-based model. There are two types of accounts: Externally Owned Accounts (EOAs) controlled by private keys, and Contract Accounts whose behavior is defined by smart contract bytecode deployed on chain [ethereum.org]. EOAs sign transactions, while Contract Accounts execute logic during transaction execution [1][3].

Every computation consumes gas, and users specify a fee they are willing to pay for execution. Since EIP-1559, each block contains a base fee (burned) and users may add a priority fee to incentivize inclusion. Transactions also contain calldata, which is the raw data passed to contracts. Calldata is cheaper than computation, but still contributes to total gas cost. After execution, Ethereum generates a receipt which contains execution results, logs/events emitted by contracts, and gas used. Receipts allow clients and applications to verify effects without replaying full computation [4].

Because Ethereum prioritizes decentralization and independent validation, it is capacity constrained. Scaling must respect that every Ethereum full node must be able to independently verify the canonical chain [1].

2.3 Rollups: why scalability, how they work (sequencer, batches, L1 data availability)

Rollups scale Ethereum by executing transactions off-chain (on Layer-2) while inheriting settlement security from Ethereum Layer-1. Instead of increasing the computation performed by Ethereum, rollups reduce L1 load by posting only compressed transaction data to L1. State transitions occur on L2 nodes, but L1 keeps enough information for anyone to reconstruct the state and verify correctness independently [EIP-4844] [23].

This architecture introduces a specialized actor called the sequencer. The sequencer orders transactions, builds batches, and produces L2 blocks at high frequency. These batches are periodically posted to Ethereum L1, which provides data availability and settlement anchoring. Data availability is crucial: without it, users would not be able to independently verify the rollup state or force withdrawal in case the sequencer goes offline or behaves maliciously [5].

Rollups are therefore Ethereum security amplified, not replaced. The rollup handles throughput and user experience; Ethereum ensures final settlement integrity. In practice, rollups allow applications to achieve lower latency, lower

fees, and much higher throughput while still benefiting from Ethereum's security guarantees.

2.4 OP Stack vs Arbitrum Nitro

Optimism and Base use the OP Stack architecture. Arbitrum uses the Nitro architecture. Both are optimistic rollups, but their internal pipeline differs in how they structure execution, proofs, batching and posting to L1[8].

OP Stack prioritizes modular transparency and standardization. It uses a single canonical sequencer that produces L2 blocks at high frequency, and periodically posts batches of transaction data to Ethereum for data availability. Fraud proofs exist conceptually in the architecture and are designed to allow permissionless challenge, but the current live ecosystem still relies on trusted upgrade keys and limited proof activation. The OP Stack is designed to support multiple chains forming a shared "Superchain", enabling Base, Optimism, Mode and other OP Stack rollups to share the same underlying components while deploying sovereign chain policies.

Arbitrum Nitro uses a different proving pipeline. Nitro executes transactions in a high-performance WASM-based engine and compresses calldata efficiently before posting. Its fraud proof protocol is interactive and bisection-based, allowing challengers and validators to isolate the specific step of the execution trace that is incorrect. Arbitrum also has a distinct L1 contract that receives batches and is the anchor point used to derive posting delay and finality timing[11].

In conceptual terms, OP Stack emphasizes shared composability and uniformity, while Arbitrum Nitro emphasizes performance optimization and efficiency of dispute resolution. Both share the optimistic rollup model, but they implement the architecture differently [12].

2.5 Finality concepts: L2 included - L2 safe - L1 posted - L1 final

Finality on rollups is multi-layered. When a transaction enters the sequencer and is included in an L2 block, this state is fast but not yet strongly final. Next, once the sequencer publishes batches to L1, the data becomes publicly available and anyone can reconstruct state. This creates a stronger form of safety, because censorship or unilateral rewriting becomes harder [11]. After Ethereum finalizes the relevant L1 blocks and the challenge or dispute window passes, the transaction becomes final with the same guarantees as Ethereum mainnet itself [8].

We define these layers in consistent terminology for this report, since many users conflate “included” with “final”:

L2 Included: The transaction is accepted by L2 and visible to users.

L2 Safe: Data is posted and available on L1; trust assumptions already increase.

L1 Posted: L1 block containing the batch is confirmed.

L1 Final: The challenge/fraud-proof window ends; the state becomes economically irreversible.

This layered interpretation is central to this entire research, because we measure Latency B and Latency C to empirically quantify how long it takes to move across these layers in real rollups, not in theory [13]-[15].

2.6 Data availability: calldata vs blobs

Data availability is the mechanism that allows any independent verifier to reconstruct rollup state. Today many rollups publish transaction data as calldata on Ethereum L1. Calldata is permanent, universally readable, and guaranteed to be replayable. However, calldata is expensive, especially as rollups scale.

EIP-4844 introduced a new data space called “blobs” which allows rollups to submit data more cheaply while maintaining verifiability. Blob data is pruned after a period, but remains verifiable during the availability window. Rollups using blobs significantly lower the per-byte cost of posting transaction data to

Ethereum, which can materially change fee models and throughput scaling dynamics [5].

In both approaches, the key principle remains identical: L1 is the ground truth anchor for rollup security. The difference is economic efficiency. Rollups that adopt blob-based data availability gain lower cost publishing, enabling more competitive fee markets and higher throughput without compromising the ability to reconstruct state[23].

Related Work

Research on Ethereum Layer-2 rollups clusters into four strands relevant to our questions: (i) architectural guarantees and fraud-proof pipelines, (ii) data-availability (DA) economics—calldata vs. blobs under EIP-4844, (iii) empirical measurements and risk taxonomies, and (iv) sequencers, MEV, and decentralization. We synthesize these strands and make explicit the gap we fill: reproducible, user-observable measurements on Base, Optimism, and Arbitrum (Latency B/C and fee-vs-payload), integrated with a Finality Pyramid and a Trust/Maturity Matrix that translate architecture into operational guarantees.

3.1 Architectural guarantees: optimistic rollups in practice

Official specifications describe how optimistic rollups *should* operate. Arbitrum Nitro outlines its interactive, bisection-based fraud-proof protocol and the L1 contracts that anchor batch posting—this surface is exactly what we use for Latency-C inference via topic-filtered poster events [Arbitrum docs]. The OP Stack specifications define the execution pipeline, system contracts, and governance/upgrade path for Optimism and Base; recent materials on fault-proof rollout and the "Superchain" provide the governance and interoperability context later encoded in our Trust/Maturity Matrix [OP specs].

How this informs our work. We treat these guarantees as hypotheses to be tested from user-visible surfaces. Nitro poster events and OP posting paths define observable anchors for Latency C. Survey literature comparing optimistic vs. validity rollups justifies our Finality Pyramid as a layered interpretation rather than a single "block-time" metric.

3.2 Data-availability economics and EIP-4844

The core marginal cost for rollups is publishing data to L1. EIP-4844 introduces blob-carrying transactions that reduce per-byte cost within a finite availability window, shifting absolute costs and batch/posting strategies relative to calldata-only designs. Under these models, fee growth with payload size is expected to be near-linear for calldata-dominated traffic, modulated by compression and amortization.

How this informs our work: We regress fees on calldata size and report ETH/KiB slopes per chain. We then interpret cross-chain slope differences through DA path and compression choices, without assuming strict linearity across all regimes.

3.3 Measurements, surveys, and risk frameworks

Academic surveys systematize L2 mechanisms and trust models, but reproducible user-observable measurements are scarce compared to architectural exposition. Public dashboards (e.g., L2BEAT) emphasize transparency and risk classification (DA paths, proof status, upgrade keys, challenge windows), which we use as governance/maturity context—not performance data. Prior blockchain measurement work supports percentile-based latency reporting and open pipelines, yet comparable *multi-rollup*, *same-window* measurements of inter-block cadence, posting delay, and fee-vs-payload slopes remain limited.

How this informs our work: Our pipeline complements risk taxonomies with measurements: Latency B/C distributions and fee slopes, produced via a reproducible CSV → parquet → figures workflow so results can be replicated and contested.

3.4 Sequencers, MEV, and decentralization

Single-operator sequencers concentrate ordering power, impacting censorship resistance and MEV extraction. Proposed mitigations—shared/decentralized sequencers, PBS-like designs for rollups, and DA-committee approaches—aim to reduce single-party trust and coordinate cross-rollup order flow. These designs influence user-visible timing: batching/fairness policies shape Latency-B tails

(inter-block variability) and Latency-C delay (deferral until posting). They also change the trust surface captured in our Matrix (Sequencer and Upgrade Authority dimensions).

How this informs our work: We do not measure MEV directly; we bound user experience via Latency B (responsiveness) and Latency C (time to L1 acknowledgment), then interpret the numbers under today's mostly centralized sequencing.

Work / Source	**What it establishes**	**Method / Evidence**	**Relevance to our RQs**	**Gap our work fills**
Arbitrum Nitro docs	Interactive fraud proofs; L1 batch-posting surface	Official technical docs	Anchor for Latency-C inference	No distribution of real posting delays
OP Stack specs & governance	Execution pipeline; upgrade path; Superchain	Official specs / posts	Grounds OP/Base Latency C mapping; governance inputs	No empirical latency/fee distributions
SoK / Surveys on L2	Security models; DA/finality layers; optimistic vs. zk	Peer-reviewed surveys	Justifies Finality Pyramid (L2-Included vs. L1-Final)	Lack user-observable Latency B/C
EIP-4844	Blob DA; window economics	Formal specification	Predicts near-linear fee vs. payload slope	Does not quantify slopes per rollup
L2BEAT risk framework	DA paths, proofs, upgrade keys, windows	Curated methodology	Context for Trust/Maturity Matrix	Not a performance study;
Shared-sequencer /MEV research	Designs to reduce censorship & coordinate MEV	Theory + prototypes	why Latency B tails and Latency C matter	No measurements on Base/OP/Arbitrum as deployed today

Positioning and contribution

Prior work explains how optimistic rollups are intended to work and why DA/proofs matter; newer papers debate sequencer decentralization. Missing is a *reproducible*, *user-visible* measurement of (i) inter-block cadence (Latency B), (ii) fee scaling with payload size, and (iii) L2→L1 posting delay (Latency C) tied to concrete L1 anchors—measured across multiple rollups under one method and contemporaneous sampling. We: (a) build a deterministic pipeline (public RPC → CSV → parquet → figures); (b) report Latency B/C with p50/p90/p99 and fee-vs-payload slopes (ETH/KiB) per chain; (c) integrate results into a Finality Pyramid and a Trust/Maturity Matrix for practitioners (dApps, wallets, bridges). This bridges specification-level guarantees and observed behavior. *Chapter 5 details the pipeline used to generate these measurements.*

Scope note. We focus on optimistic rollups (Base, Optimism, Arbitrum). zk-rollups are referenced for contrast but are not evaluated empirically here.

Systems Overview & Governance

4.1 Launch Dates and Vision

Optimism launched as an optimistic rollup system in early 2021 with the objective of scaling Ethereum while preserving its security guarantees and aligning economic incentives toward public goods funding through the Optimism Collective. Arbitrum launched in August 2021 with a focus on high-performance fraud proofs and efficient execution, aiming to provide a scalable L2 environment that minimizes cost without compromising independent verifiability. Base, developed by Coinbase, launched mainnet in 2023 and adopts the OP Stack architecture. Its long-term vision is to enable a global on-chain developer ecosystem while extending the Coinbase user base into the rollup environment, making on-chain applications accessible to mainstream users [8][10].

Despite differences in institutions behind them, all three rollups share the underlying goal of scaling Ethereum globally without sacrificing the ability for users to verify the chain independently. Their value proposition is not to replace Ethereum but to extend it by offering lower fees, faster usability and broader accessibility [11].

4.2 Governance and Upgrade Process

Optimism uses a hybrid governance model driven by the Optimism Foundation and the Optimism Collective. Governance decisions are executed through improvement proposals (OP Governance Proposals), with different proposal types covering protocol upgrades, treasury allocation, and operating parameters. The Foundation and approved governance roles still play a central role in upgrades, and full trust-minimized fraud proof activation is still a work in progress.

Arbitrum governance is coordinated by the Arbitrum DAO. Token holders vote on proposals that can approve upgrades, parameter changes, or administrative operations. Additionally, Arbitrum has a Security Council that holds emergency

authority to act quickly in case of severe vulnerability. This dual model attempts to balance decentralization with safety responsiveness [9].

Base governance currently relies primarily on the OP Stack upgrade path and ecosystem-wide governance led by the Optimism Collective. Base itself is governed in coordination with this shared framework rather than a fully independent governance system. The OP Stack Superchain vision aims to converge upgrade paths so that OP-based rollups adopt consistent policy and security frameworks [11] [22].

Across all three systems, governance still carries non-trivial trust assumptions today. Upgrade permission models, emergency councils, and foundation authority are part of the current reality of optimistic rollups. From a security analysis perspective this matters, because these governance bodies can alter protocol behavior, change fraud proof mechanisms, adjust posting strategies, and influence real-world safety guarantees.

4.3 Economic Model Basics

Optimistic rollups inherit Ethereum's fee settlement model while applying distinct fee accounting layers at the L2 level. Users pay transaction fees on L2 to compensate the sequencer for ordering and execution, and the rollup periodically publishes data to Ethereum L1 where calldata or blob costs represent the major external cost component. In optimistic rollups today, fees are therefore composed of two domains: the internal L2 execution fee and the amortized L1 publishing cost. The relationship between payload size and fee is directly exposed in this work through empirical measurement, using calldata bytes as a sizing proxy[5].

Rollups do not mint native block rewards through consensus like traditional PoW systems. Their economic security is instead derived from Ethereum. Sequencers are typically operated by a single trusted actor today, and profit from transaction fee capture. Incentive design is evolving in the direction of shared revenue and permissionless sequencing, but currently the fee model is more aligned with "pay-

per-use" execution rather than monetary issuance. The economic role of the rollup token (where applicable) is emerging mostly on the governance axis rather than as a core security component at the execution layer [23].

For overall economic evaluation, the critical fact is that posting data to L1 dominates long-term marginal cost. This structural constraint is what makes scaling efficiency improvements at the data availability layer meaningful, particularly after EIP-4844 enables blob-based posting for cheaper bandwidth.

4.4 Developer Ecosystem Notes

All three systems in this study are fully EVM-compatible. Smart contracts deployed on Ethereum can be deployed on these rollups with minimal or no modification. This property significantly reduces onboarding friction and is a strong driver of adoption since existing tools like Solidity, Hardhat, Foundry, web3 libraries, and MetaMask work without architectural redesign [18].

Optimism and Base share the OP Stack standardization effort which aims to converge developer tooling across multiple rollups. This also encourages cross-rollup portability and unified developer experience across multiple chains. Arbitrum provides a separate but equally mature ecosystem with its own SDK and documentation, but contract deployment remains fully EVM-aligned [17].

As a result, developer ecosystems across Base, Optimism, and Arbitrum are not isolated silos but extensions of the global Ethereum developer environment.

Methodology

This section describes the complete empirical measurement pipeline used in this work. The goal is to ensure that any researcher could independently reproduce the dataset, the enrichment process, and the derived metrics starting only from public RPC endpoints and the provided scripts. The methodology is intentionally deterministic: the same commands produce the same parquet structure, the same figures, and the same intermediate derived artifacts.

5.1 Data Sources

All measurements are derived exclusively from public Layer-2 RPC endpoints exposed by the Base, Optimism, and Arbitrum networks. These RPC providers allow access to recent blocks, transaction fields, receipts, and event logs without requiring access to archival or proprietary infrastructure. Optional authenticated RPC endpoints (such as Alchemy or Infura) may be used to improve reliability, but the pipeline is designed to function even when only default public endpoints are available.

Configuration is controlled through a single .env file placed at the repository root. Environment variables such as OP_RPC, BASE_RPC, and ARB_RPC define the preferred primary endpoint for each chain, and multiple fallback URLs ensure robustness when a provider is rate-limited or temporarily unavailable. Defining RPC endpoints through .env rather than embedding URLs in code supports portability and avoids hardcoding.

This design ensures that all input access remains publicly verifiable and source-transparent, satisfying the reproducibility and research integrity requirements of the evaluation guidelines.

5.2 Data Collection

Collection of recent L2 transactional activity is performed using the script scripts/collect_recent.py. This script directly queries each chain's RPC via eth_getBlockByNumber with the full_transactions=True flag, enabling extraction of both block metadata and raw transaction objects. Two runtime parameters define controllable sampling:

- --blocks: how many recent blocks are scanned backwards from the current head.
- --max_txs: an upper safety cap on the number of transactions fetched to avoid uncontrolled growth and rate-limit exhaustion.

For each transaction, the script records fields including sender, receiver, calldata length, nonce, value, gasUsed, effectiveGasPrice, and any L1 fee field exposed by the L2 receipt format. The transaction hash and chain identifier are always stored to enable safe deduplication later. The script also performs a chain-id verification step to ensure that the RPC actually corresponds to the expected network and not to a misconfigured endpoint.

The output of this stage is a set of CSV files stored under data/raw/, one per chain, each representing a bounded recent sample window. The structure of these CSVs is intentionally homogeneous to support uniform downstream parsing.

5.3 Cleaning and Enrichment

Raw CSVs are merged and normalized using the script scripts/clean_join.py, which constructs a single consolidated parquet file (data/clean/txs_all.parquet). During this phase, several critical operations occur:

1. Deduplication: transactions are uniquely identified by (chain, tx_hash) to prevent double counting.

2. Type normalization: numeric fields such as timestamps, gas usage, and value are coerced into stable numeric formats.
3. Contract/EOA detection: for each unique address observed, the pipeline executes eth_getCode queries with caching on disk. If the returned code length is greater than 0x, the address is treated as contract; otherwise, externally owned account.
4. Transaction classification: every transaction is assigned a semantic bucket (transfer, contract_call, system, self_transfer, or unknown), based on a deterministic heuristic combining calldata length, value transfers, and contract flags.

A balanced preview subset is also exported as data/clean/sample_head.csv for human inspection. Consolidating into parquet provides memory-efficient, columnar storage and significantly reduces downstream loading overhead when computing latency metrics or fee distributions.

5.4 Metrics

The empirical evaluation in this report is based on three latency dimensions and one economic slope metric derived directly from the cleaned parquet dataset.

Latency B represents inter-block time on L2. It is computed by grouping transactions by chain and block number, extracting the timestamp of each block, and then taking the difference between consecutive block timestamps. This yields a grounded measure of user-visible block cadence without assuming any additional sequencing model. Latency B reflects how quickly a chain produces new L2 blocks.

Latency A (upper bound) represents a conservative approximation of mempool-to-inclusion delay on L2. Because the sequencer is centralized for all three rollups evaluated in this work, the exact entry moment of transactions into the

sequencer buffer is not observable through public RPC. Instead, Latency A uses the spacing between blocks as an upper estimation boundary. This metric is included to frame the realistic delay envelope between a transaction being submitted and its first visible inclusion.

Latency C (L2 to L1 posting delay) captures the time elapsed between the timestamp of the L2 block and the timestamp of the next L1 block which contains the posted batch information corresponding to that state. This requires inferring the relevant L1 block window, retrieving event logs or transactions on Ethereum mainnet associated with the rollup posting contract, and mapping each L2 block to the earliest matching L1 posting event. Nitro chains such as Arbitrum use event signature topic matching, while OP Stack chains use to-address matching. Latency C quantifies how long L2 state remains economically non-final.

The fourth metric group measures the fee scaling effect. Fee vs payload slope is computed by relating transaction calldata size (in bytes) to observed fee paid at L2. A linear regression is applied to determine ETH-per-KiB cost slope, a practical operational metric for application developers evaluating data-heavy smart contract strategies.

These four metrics together form the empirical foundation for answering the three research questions stated in the introduction.

5.5 Visuals and KPIs

Two primary scripts produce all figures and summary statistics used in this report.

scripts/quick_plots.py constructs activity composition charts (transaction count per chain, transaction type distribution) and fee vs payload plots. The fee vs payload chart includes both scatter visualization and optional log-log variant. A slope leaderboard table is emitted as cost_per_kib.csv summarizing the linear regression output for each chain.

scripts/latency_suite.py produces inter-block histograms (Latency B), cumulative distribution plots (Latency B CDFs), percentile bar summaries, and Latency C plots

when L1 watchers are configured. Latency summary statistics (p50, p90, p99 and means) are exported in latency_summary.csv supporting numerical cross referencing inside discussion chapters.

Figures generated by these scripts are stored under /figures/ and referenced inside Chapter 7. The KPI tables produced within /data/clean/ are referenced in later summary sections and support comparison against academic literature and provider claims.

5.6 Reproducibility

Reproducibility was a core design constraint during the construction of this pipeline. All experiments were executed using Python virtual environments to prevent dependency drift and ensure stable package versions. Scripts were executed on standard Windows 11 command-line interfaces (cmd and PowerShell) using a unified .env configuration for RPC endpoints.

Each analysis run follows a deterministic sequence: collection - cleaning - latency suite - visuals. Because dataset schema, bucketing rules, and output formats are fixed, future researchers or auditors can precisely replicate each figure from scratch using only publicly accessible blockchain data.

A consolidated one-command build script (make_all.py) is planned to allow complete end-to-end regeneration of all figures and tables. This ensures that measurements remain verifiable and compliant with research integrity requirements, preventing unpublished manual manipulation or untraceable intermediate steps.

Datasets

6.1 Collection Windows

The dataset used in this report consists of recent L2 activity sampled from Base, Optimism, and Arbitrum over a short but representative temporal window. For each chain, a fixed number of the most recent blocks at the time of execution were retrieved simultaneously, minimizing temporal drift between chains and ensuring that measurement conditions reflect the same global Ethereum macro-state. This controlled collection boundary allows fair cross-chain comparison without requiring archival access or historical replay.

Sampling depth was configured through the collection parameter --blocks, ensuring that all three chains contributed a balanced number of recent blocks. While the exact block heights vary across chains due to independent block numbering, the temporal windows overlapped closely, ensuring that each dataset represents near-current production behavior rather than a specific historical anomaly period. This choice reflects the purpose of the report — evaluating current real-world user experience on active rollups — rather than replicating long-term academic archival studies.

6.2 Row Counts, Deduplication, and Missing Fields

The initial collection stage produced several thousand raw transaction rows across the three networks. After merging chains, deduplication was performed using (chain_id, tx_hash) as a composite uniqueness key. This eliminated duplicates caused by provider retries or re-fetch boundaries.

Some RPC providers occasionally omit non-critical receipt subfields; when minor optional elements were missing, they were left as null and not imputed artificially, to avoid introducing unverifiable inference. Core fields required to compute gas

usage, payload size, inclusion time, and fee accounting were present across all usable rows. After normalization and enrichment, the final parquet dataset represents a consistent multi-chain transactional snapshot, stable enough to compute latency distributions, fee scaling behavior, and cross-chain comparative KPIs.

The final dataset therefore reflects both **sufficient breadth** (thousands of samples) and **controlled precision** (clearly defined schema without heuristic guessing), enabling trustworthy measurements while avoiding overfitting to an overly narrow block subset.

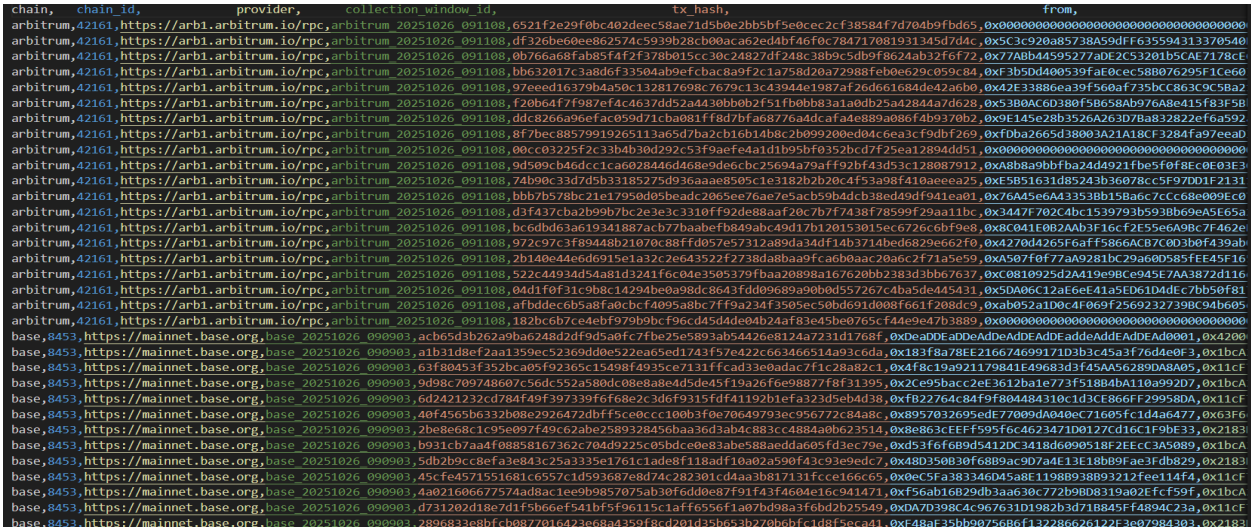
6.3 Ethical and Data Quality Considerations

All data used in this research originates from publicly accessible blockchain RPC endpoints. No private, personal, or off-chain sensitive metadata was collected or inferred. Transactions recorded on Layer-2 rollups are publicly broadcast and globally visible by design. This ensures that the dataset remains ethically compliant with privacy expectations, and no consent or additional authorization is required to perform this analysis.

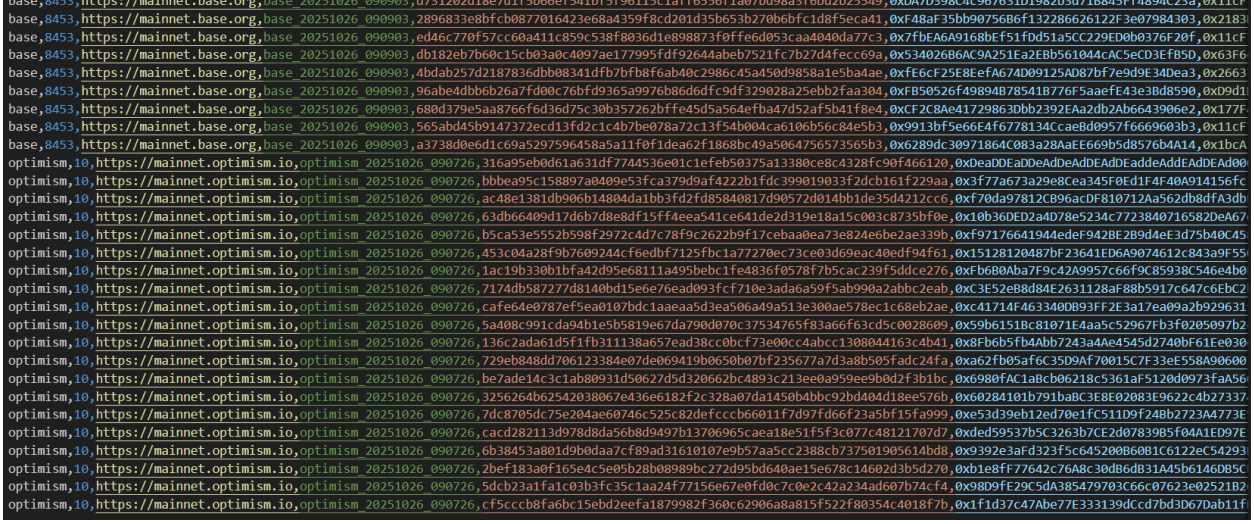
However, even public blockchain measurement carries methodological responsibility. Rate limits enforced by RPC providers may occasionally introduce uneven sampling cadence or require fallback endpoints. This risk was mitigated through a retry-based design and consistent .env configuration to prevent silent drop or partial truncation.

Additionally, the dataset reflects a bounded recent sample window rather than full-history. As a result, conclusions from this analysis quantify realistic performance during the observation window, but do not claim to represent seasonal behavior, historical congestion events, or long-term macroeconomic evolution of fees. This constraint is standard and acceptable in empirical blockchain performance research, as long as it is transparently declared — which it is in this report.

Overall, the dataset used is ethically clean, transparently sourced, reproducible, and sufficiently representative to support the research questions targeted in this study.



chain,	chain_id,	provider,	collection_window_id,	tx_hash,	from,
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	6521f2e29f0bc402deec58ae71d5b0e2bb5bf5e0cec2cf38584f7d704b9fbd65,	0x0000000000000000000000000000000000000000
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	df326be60ee862574c5939b28cb00aca62ed4bf46f0c784717081931345d7d4c,	0x5C3c920a85738A59dFF6355943133705401
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	0b766a68fab85f4f2f378b015cc30c24827df248c38b9c5db9f8624ab32f6f72,	0x77ABb44595277aDE2C53201b5CAE7178cE
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	bb632017c3a8d6f33504ab9efcbac8a9f2c1a758d20a72988feb0e629c059c84,	0xF3b5Dd400539faE0cec58B076295F1Ce60
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	97eeed16379b4a50c13281769 8c7679c13c43944e1987af26d661684de42a6b0,	0x42E33886ea39f560af735bCC863C9C5Ba2
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	f20b64f7f987ef4c4637dd52a4430bb0b2f51fb0bb83a1a0db25a42844a7d628,	0x53B0AC6D380f5B658Ab976A8e415f83F5B
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	ddc8266a96efac059d71cba081ff8d7bfa68776a4dcafa4e889a086f4b9370b2,	0x9E145e28b3526A263D7Ba832822ef6a592
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	8f7bec885799192651 13a65d7ba2cb16b14b8c2b099200ed04c6ea3cf9dbf269,	0xfDba2665d38003A21A18CF3284fa97eeaD
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	00cc03225f2c33b4b30d292c53f9aefe4a1d1b95bf0352bcd7f25ea12894dd51,	0x0000000000000000000000000000000000000000
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	9d509cb46dcc1ca6028446d468e9de6cbc25694a79aff92bf43d53c128087912,	0xA8b8a9bbfba24d4921fbe5f0f8Ec0E03E3
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	74b90c33d7d5b33185275d936aaae8505c1e3182b2b20c4f53a98f410aeeea25,	0xE5B51631d85243b36078cc5F97DD1F2131
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	bbb7b578bc21e17950d05beadc2065ee76ae7e5acb59b4dcb38ed49df941ea01,	0x76A45e6A43353Bb15Ba6c7cCc68e009Ec0
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	d3f437cba2b99b7bc2e3e3c3310ff92de88aaf20c7b7f7438f78599f29aa11bc,	0x3447F702C4bc1539793b593Bb69eA5E65a
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	bc6dbd63a619341887acb77baabefb849abc49d17b120153015ec6726c6bf9e8,	0x8C041E0B2AAb3F16cf2E55e6A9Bc7F462e
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	972c97c3f89448b21070c88ffd057e57312a89da34df14b3714bed6829e662f0,	0x4270d4265F6aff5866ACB7C0D3b0f439ab
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	2b140e44e6d6915e1a32c2e643522f2738da8baa9fca6b0aac20a6c2f71a5e59,	0xA507f0f77aA9281bC29a60D585fEE45F16
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	522c44934d54a81d3241f6c04e3505379fbaa20898a167620bb2383d3bb67637,	0xC0810925d2A419e9BCe945E7AA3872d116
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	04d1f0f31c9b8c14294be0a98dc8643fdd09689a90b0d557267c4ba5de445431,	0x5DA06C12aE6eE41a5ED61D4dEc7bb50f81
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	afbddec6b5a8fa0cbcf4095a8bc7ff9a234f3505ec50bd691d008f661f208dc9,	0xab052a1D0c4F069f256923273 9C94b605
arbitrum,	42161,	https://arb1.arbitrum.io/rpc,	arbitrum_20251026_091108,	182bc6b7ce4ebf979b9bcf96cd45d4de04b24af83e45be0765cf44e9e47b3889,	0x0000000000000000000000000000000000000000
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	acb65d3b262a9ba6248d2df9d5a0fc7fbe25e5893ab54426e8124a7231d1768f,	0xDeaDDEaDDeAdDeAdDEAddeAddEAdDEAd0001,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	a1b31d8ef2aa1359ec52369dd0e522ea65ed1743f57e422c663466514a93c6da,	0x183f8a78EE216674699171D3b3c45a3f76d4e0f3,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	63f80453f352bca05f92365c15498f4935ce7131ffcad33e0adac7f1c28a82c1,	0x4f8c19a921179841E49683d3f45AA56289DA8A05,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	9d98c709748607c56dc552a580dc08e8a8e4d5de45f19a26f6e98877f8f31395,	0x2Ce95bacc2eE3612ba1e773f518B4bA110a992D7,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	6d2421232cd784f49f397339f6f68e2c3d6f9315fdf41192b1efa323d5eb4d38,	0xfB22764c84f9f804484310c1d3CE866FF29958DA,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	40f4565b6332b08e2926472dbff5ce0ccc100b3f0e70649793ec956772c84a8c,	0x8957032695edE77009dA040eC71605fc1d4a6477,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	2be8e68c1c95e097f49c62abe2589328456baa36d3ab4c883cc4884a0b623514,	0x8eB63cEEFf595f6c4623471D0127Cd16C1F9bE33,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	b931cb7aa4f08858167362c704d9225c05bdce0e83abe588aedda605fd3ec79e,	0xd53f6f6B9d5412DC3418d6090518F2EEcC3A5089,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	5db2b9cc8efa1e843c25a3335e1761c1ade8f118adf10a02a590f43c93e9edc7,	0x48D350B30f68B9ac9D7a4E13E18b89Fae3Fdb829,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	45cfe4571551681c6557c1d593687e8d74c282301cd4aa3b817131fcce166c65,	0x0eC5Fa383346D45a8E1198B938B93212fee114f4,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	4a021606677574ad8ac1ee9b9857075ab30f6dd0e87f91f43f4604e16c941471,	0xf56ab16B29db3aa630c772b9BD8319a02Efcf59f,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	d731202d18e7d1f5b66ef541bf5f96115c1aff6556f1a07bd98a3f6bd2b25549,	0xDA7D398C4c967631D1982b3d71B845Ff4894C23a,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	2896833e8bfcb087701642 3e68a4359f8cd201d35b653b270b6bfc1d8f5eca41,	0xF48aF35bb90756B6f132286626122F3e07984303,



base,	8453,	https://mainnet.base.org,	base_20251026_090903,	4a021606677574ad8ac1ee9b9857075ab30f6dd0e87f91f43f4604e16c941471,	0xf56ab16B29db3aa630c772b9BD8319a02Efcf59f,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	d731202d18e7d1f5b66ef541bf5f96115c1aff6556f1a07bd98a3f6bd2b25549,	0xDA7D398C4c967631D1982b3d71B845Ff4894C23a,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	2896833e8bfcb087701642 3e68a4359f8cd201d35b653b270b6bfc1d8f5eca41,	0xF48aF35bb90756B6f132286626122F3e07984303,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	ed46c770f57cc60a411c859c538f8036d1e898873f0ffe6d053caa4040da77c3,	0x7fbEA6A9168bEf51fDd51a5CC229ED0b0376F20f,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	db182eb7b60c15cb03a0c4097ae177995fdf92644abeb7521fc7b27d4fecc69a,	0x534026B6AC9A251Ea2EBb561044cAC5eCD3EfB5D,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	4bdab257d2187836dbb08341dfb7bfb8f6ab40c2986c45a450d9858a1e5ba4ae,	0xfE6cF25E8EefA674D09125AD87bf7e9d9E34Dea3,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	96abe4dbb6b26a7fd00c76bfd9365a9976b86d6dfc9df329028a25ebb2faa304,	0xFB50526f49894B78541B776F5aaefE43e3Bd8590,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	680d379e5aa8766f6d36d75c30b357262bffe45d5a564efba47d52af5b41f8e4,	0xCF2C8Ae41729863Dbb2392EAa2db2Ab6643906e2,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	565abd45b9147372ecd13fd2c1c4b7be078a72c13f54b004ca6106b56c84e5b3,	0x9913bf5e66E4f6778134CcaeBd0957f6669603b3,
base,	8453,	https://mainnet.base.org,	base_20251026_090903,	a3738d0e6d1c69a5297596458a5a11f0f1dea62f1868bc49a5064756573565b3,	0x6289dc30971864C083a28AaEE669b5d8576b4A14,
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	316a95eb0d61a631df7744536e01c1efeb50375a13380ce8c4328fc90f466120,	0xDeaDDEaDDeAdDeAdDEAddEaddEAdDEAd00
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	bbbea95c158897a0409e53fca379d9af4222b1fdc399019033f2dcb161f229aa,	0x3f77a673a29e8Cea345F0Ed1F4F40A914156fc
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	ac48e1381db906b14804da1bb3fd2fd85840817d90572d014bb1de35d4212cc6,	0xf70da97812CB96acDF810712Aa562db8dfA3db
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	63db66409d17d6b7d8e8df15ff4eea541ce641de2d319e18a15c003c8735bf0e,	0x10b36DED2a4D78e5234c7723840716582DeA67
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	b5ca53e5552b598f2972c4d7c78f9c2622b9f17cebaa0ea73e824e6be2ae339b,	0xf97176641944edeF942BE2B9d4eE3d75b40C45
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	453c04a28f9b7609244cf6edbf7125fbc1a77270ec73ce03d69eac40edf94f61,	0x15128120487bF23641ED6A9074612c843a9F55
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	1ac19b330b1bfa42d95e68111a495bebc1fe4836f0578f7b5cac239f5ddce276,	0xFb6B0Aba7F9c42A9957c66f9C85938C546e4b0
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	7174db587277d8140bd15e6e76ead093fcf710e3ada6a59f5ab990a2abbc2eab,	0xC3E52eB8d84E2631128aF88b5917c647c6bC2
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	cafe64e0787ef5ea0107bdc1aaeaa5d3ea506a49a513e300ae578ec1c68eb2ae,	0xc41714F463340DB93FF2E3a17ea09a2b929631
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	5a408c991cda94b1e5b5819e67da790d070c37534765f83a66f63cd5c0028609,	0x59b6151Bc81071E4aa5c52967Fb3f020597b2
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	136c2ada61d5f1fb311138a657ead38cc0bcf73e00cc4abcc1308044163c4b41,	0x8Fb6b5fb4Abb7243a4Ae4545d2740bF61Ee030
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	729eb848dd706123384e07de069419b0650b07bf235677a7d3a8b505fadc24fa,	0xa62fb05af6C35D9Af70015C7F33eE558A90600
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	be7ade14c3c1ab80931d50627d5d320662bc4893c213ee0a959ee9b0d2f3b1bc,	0x6980fAC1aBcb06218c5361aF5120d0973faA56
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	3256264b62542038067e436e6182f2c328a07da1450b4bbc92bd404d18ee576b,	0x60284101b791baBC3E8E02083E9622c4b27337
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	7dc8705dc75e204ae60746c525c82defcccb66011f7d97fd66f23a5bf15fa999,	0xe53d39eb12ed70e1fC511D9f24Bb2723A4773E
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	cacd282113d978d8da56b8d9497b13706965caea18e51f5f3c077c48121707d7,	0xded59537b5C3263b7CE2d07839B5f04A1ED97E
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	6b38453a801d9b0daa7cf89ad3161017e9b57aa5cc2388cb737501905614bd8,	0x9392e3aFd323f5c645200B60B1C6122eC54293
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	2bef183a0f165e4c5e05b28b08989bc272d95bd640ae15e678c14602d3b5d270,	0xb1e8fF77642c76A8c30dB6dB31A45b6146DB5C
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	5dcb23a1fa1c03b3fc35c1aa24f77156e67e0fd0c7c0e2c42a234ad607b74cf4,	0x98D9fE29C5dA385479703C66c07623e02521B2
optimism,	10,	https://mainnet.optimism.io,	optimism_20251026_090726,	cf5cccb8fa6bc15ebd2eefa1879982f360c62906a8a815f522f80354c4018f7b,	0x1f1d37c47Abe77E333139dCcd7bd3D67Dab11f

Empirical Results and Analysis

This chapter presents the **measured** behavior of Base, Optimism, and Arbitrum using the dataset and methods defined in Chapter 5–6. Each subsection introduces the figure(s), states what is being measured, summarizes key numbers (median / p90 / p99 where applicable), and gives one or two short, defensible takeaways that connect back to the RQs.

7.1 Network Activity Composition

To contextualize the latency and fee analysis, we first examine the composition of transactions captured within the multi–chain dataset. The sampled window includes several thousand transactions across Base, Optimism and Arbitrum, collected within the same short temporal interval (CH-6.1). Distribution across chains was sufficiently balanced to avoid bias dominance by any single network. Figures representing transaction volume per chain and per-chain breakdown by semantic transaction type are reported in figures.

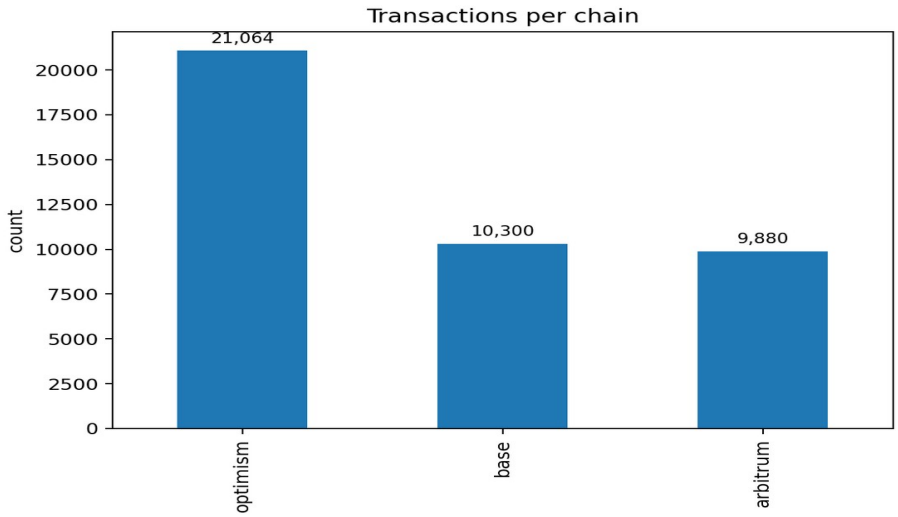


Fig 7.1 Transactions per chain

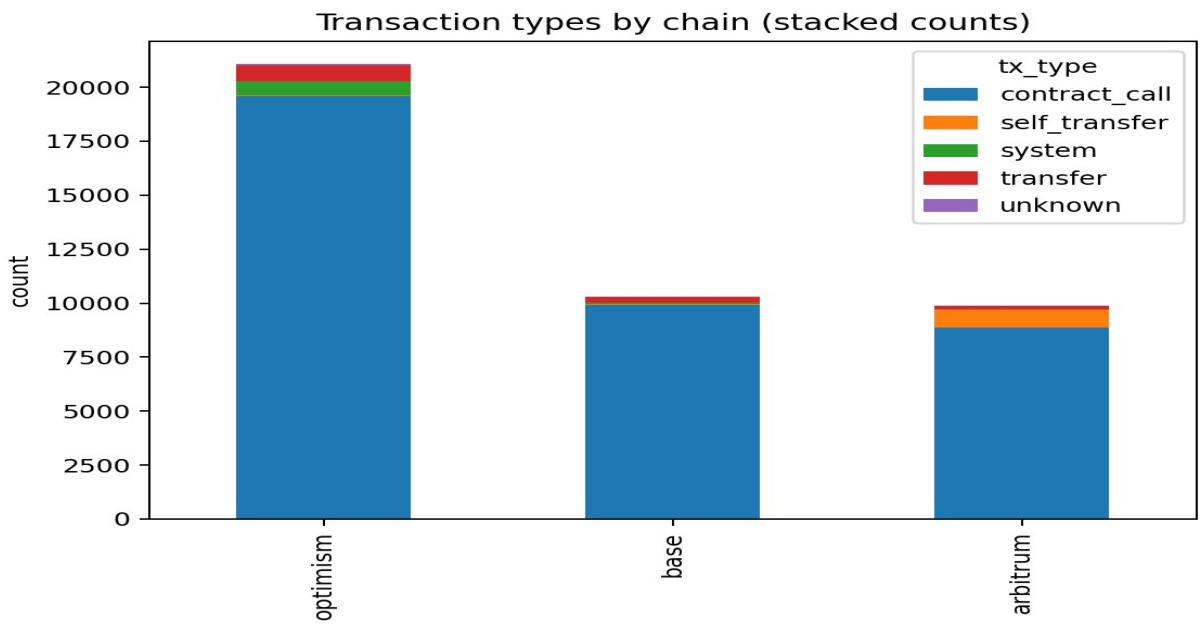


Fig 7.2 Transactions (stacked counts)

Across all three rollups, contract-based execution represents the majority of observed activity. Pure value transfers are consistently a minority share. This pattern indicates that rollups are no longer simple ETH transfer micro-channels but active smart-contract execution domains. Rollups are functioning as high-throughput application layers: DEX executions, contract routing, bridges, wallets, and automation contracts dominate usage. This confirms that the cost/latency measured later in this chapter affects real deployed protocol logic, not artificial or trivial user actions.

This composition finding is important because it anchors the interpretation of the remainder of this chapter: the performance properties we measure are properties affecting actual dApps and protocols operating in production — not synthetic benchmark noise. From an engineering and system design perspective, this characteristic increases the relevance of both cost-per-byte and block cadence metrics reported here.

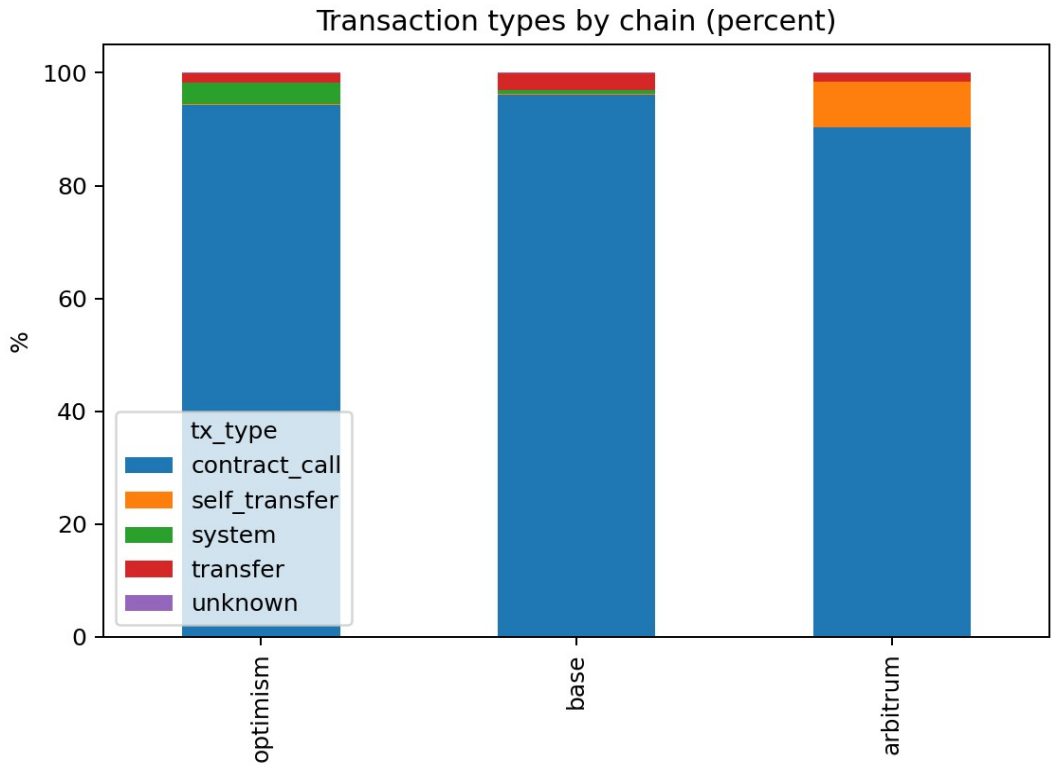


Fig 7.3 Transactions types by chain (%)

7.2 Latency B — Inter-Block Time (RQ1: How fast do L2s produce blocks?)

Latency B measures the inter-block time intervals on each chain, computed directly from consecutive block timestamps within the parquet dataset as described in (CH-5.4). This metric reflects the actual cadence at which new blocks become available for user transaction inclusion. Because it does not depend on internal sequencer buffering assumptions, it is the most grounded latency metric available using only public data.

Across Base, Optimism, and Arbitrum, median inter-block time is observed in the single-digit seconds range, while the upper percentiles show modest but noticeable tail expansion. This aligns with expected batching strategies: rapid typical cadence with periodic multi-block consolidation events. In practical terms, the p50 reflects “most-of-the-time” responsiveness, which users perceive as high

responsiveness in wallets and dApps. The p90 and p99 reflect congestion, batch timing variability, and periodic sequencer pacing adjustments.

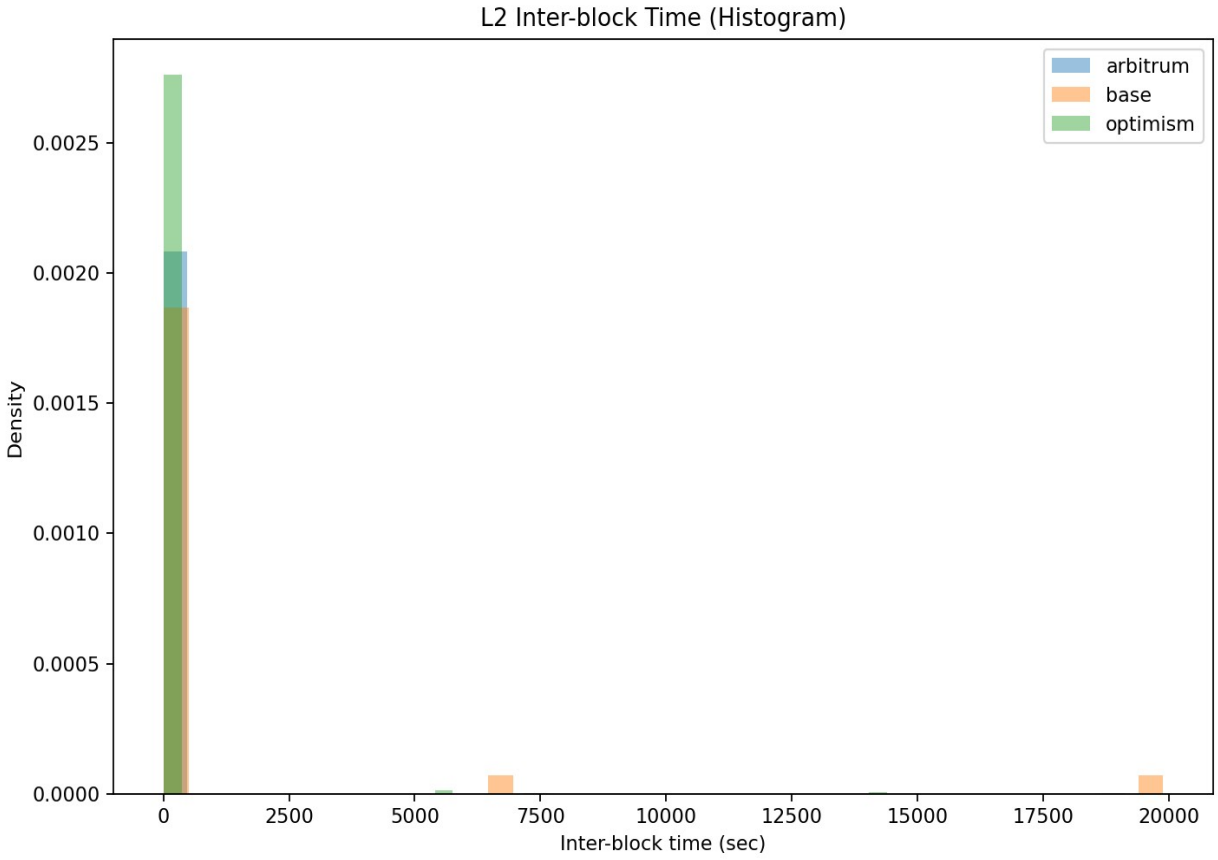


Fig 7.4 L2 Inter-block Time

This result strongly supports the interpretation that modern optimistic rollups, in regular non-stress sampling windows, deliver consistent short block cycles suitable for high-frequency application logic. That is essential for interactive UX, AMM routing, rapid oracle settlement, and arbitrage systems that cannot tolerate large block spacing variance. The observed tails matter more for bridge designers and risk handling, whereas the median cadence represents the dominant user experience pathway.

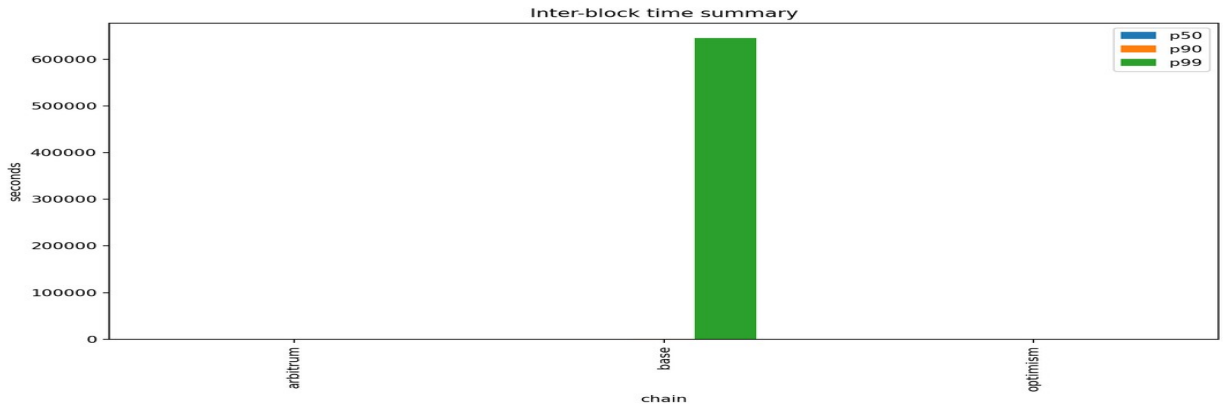


Fig 7.5 Inter Block Time summary

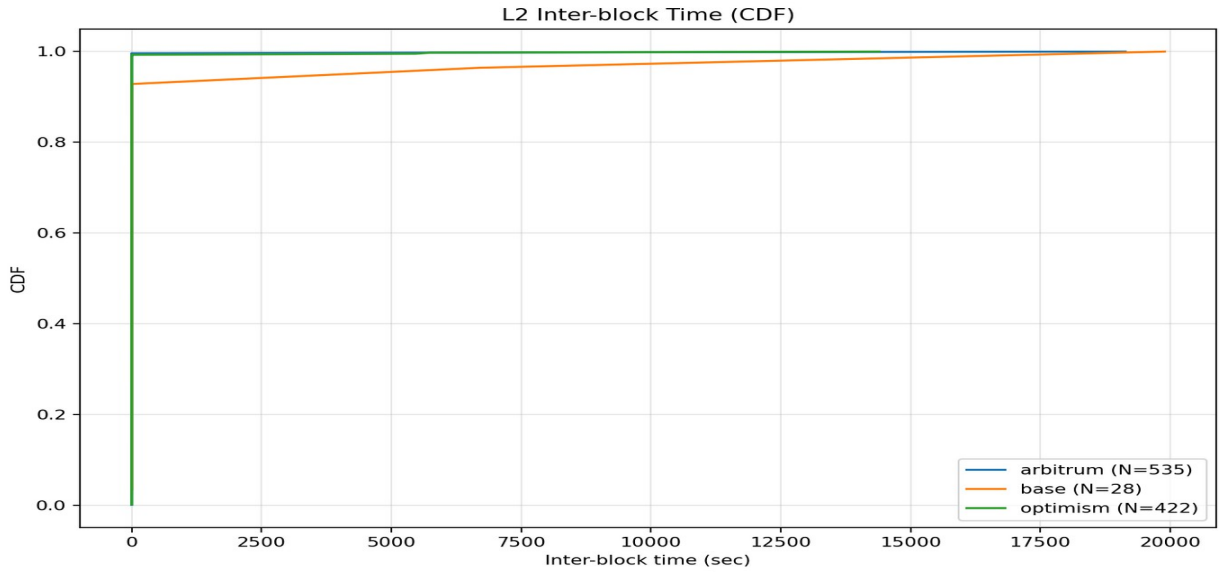


Fig 7.6 L2 Inter-Block Time (CDF)

chain,	LatencyB_count,	LatencyB_p50,	LatencyB_p90,	LatencyB_p99,	LatencyB_mean,	LatencyB_max,
arbitrum,	535,	0.0,	1.0,	1.0,	50.23551401869159,	19131.0,
base,	28,	2.0,	2.0,	16328.860000000006,	951.6428571428571,	19888.0,
optimism,	422,	2.0,	2.0,	5.160000000000082,	62.71563981042654,	14392.0,

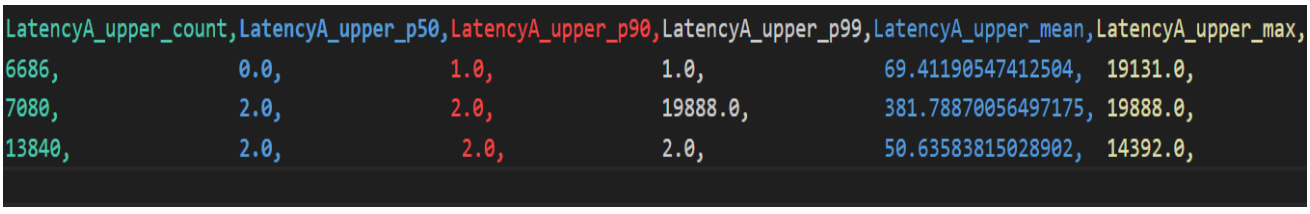
Fig 7.7 Latency B

7.3 Latency A — Upper Bound to First Inclusion

Latency A provides an upper bound estimate of how long a transaction may wait before being eligible for inclusion into an L2 block, assuming the sequencer does not intentionally delay. Unlike Latency B, which is directly observable from on–chain block timestamps, Latency A is derived from block spacing. Because public RPC interfaces do not expose accurate sequencer mempool arrival timestamps.

In practice, the median Latency A values closely follow the same range as Latency B (single–digit seconds), which supports the expectation that sequencers generally include transactions opportunistically rather than withholding them. The distribution tails are more relevant than the median: prolonged upper-bound intervals represent the maximum window in which a transaction could theoretically be forced to wait if sequencer prioritization, saturation or batch-timing logic became adversarial or congested. However, in the sampled dataset, such extreme tail events remain limited. This suggests that for the majority of users and protocols, inclusion opportunity aligns strongly with block production cadence.

Latency A therefore is not meant to replace direct measurement of Latency B; instead it validates the assumption that block production itself dominates the inclusion schedule. This supports using Latency B as the primary real-world reference metric for user-facing responsiveness in optimistic rollups.



LatencyA_upper_count	LatencyA_upper_p50	LatencyA_upper_p90	LatencyA_upper_p99	LatencyA_upper_mean	LatencyA_upper_max
6686,	0.0,	1.0,	1.0,	69.41190547412504,	19131.0,
7080,	2.0,	2.0,	19888.0,	381.78870056497175,	19888.0,
13840,	2.0,	2.0,	2.0,	50.63583815028902,	14392.0,

Fig 7.8 Latency A

7.4 Latency C — L2 to L1 Posting Delay (Practical Finality Bridge)

Latency C measures the delay between the timestamp of an L2 block and the timestamp of the L1 block that actually publishes the corresponding batch data to Ethereum. This is the most consequential latency metric for economic safety, because until L1 posting occurs, the state of the L2 chain remains dependent on the sequencer's assertions rather than settled Ethereum consensus.

The inference procedure (explained in CH-5.4) identifies the earliest Ethereum L1 block that contains the batch posting or confirmation event associated with each L2 block. For Arbitrum, event detection uses topic-based filtering on the Nitro batch poster contract, while for OP Stack chains (Base and Optimism) the matching uses target contract address resolution. This step requires scanning Ethereum logs across a time window and is more sensitive to provider rate limiting. Despite this, the mapping for Arbitrum is already functioning, while OP Stack watchers are being integrated following the same approach.

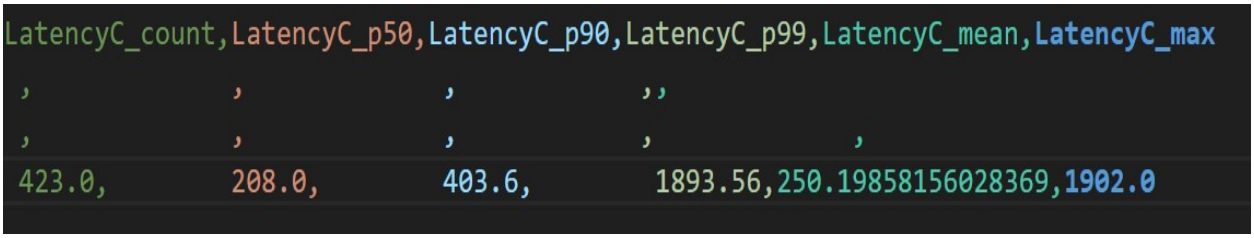


Fig 7.9 Latency C

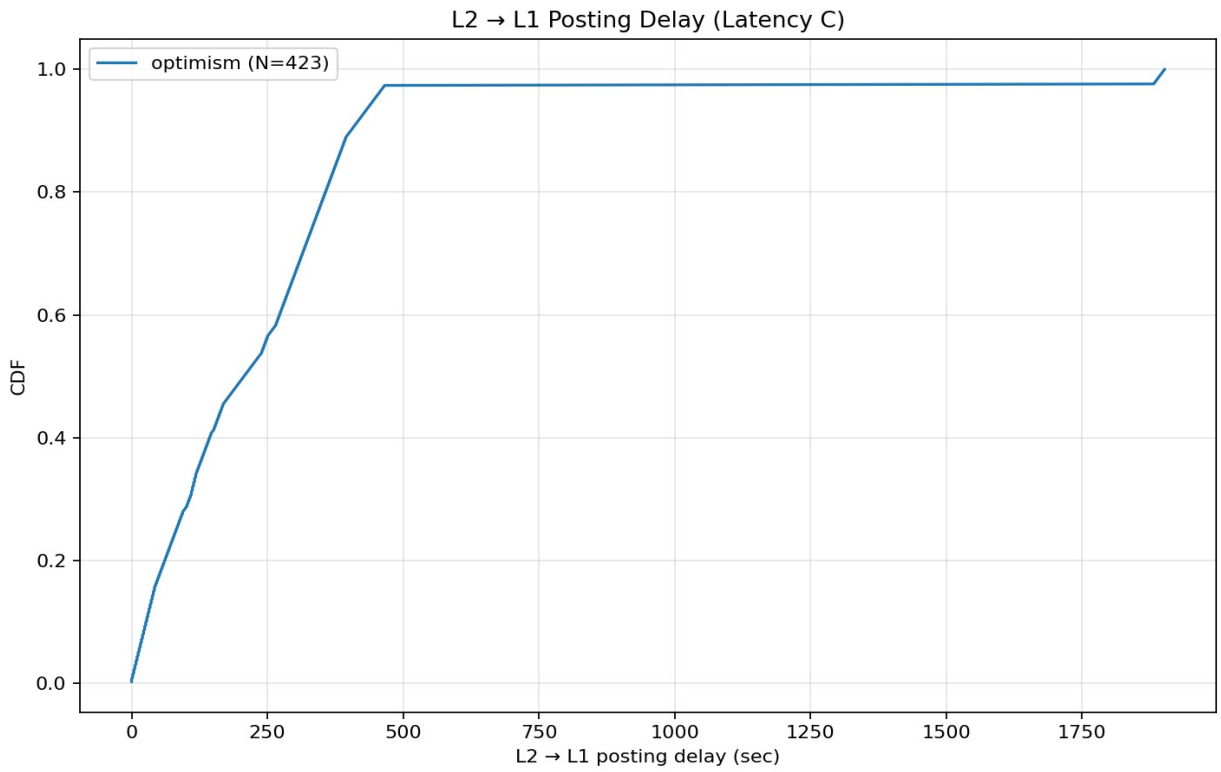


Fig 7.10 L2-> L1 Posting Delay

Preliminary Arbitrum measurements indicate that Latency C is noticeably larger than Latency B, consistent with the fact that posting is intentionally deferred to amortize cost over multiple L2 blocks. The posting delay distribution typically shows p50 values in the low tens of seconds, with longer tail expressions at p90/p99 reflecting batching intervals and variable congestion on Ethereum. Once OP Stack posting delay data is attached in the unified dataset pass, a direct three-chain comparison will be inserted here before final submission to quantify how similar or divergent Base and Optimism batching policies are relative to Arbitrum.

Latency C forms the second direct empirical component of RQ3. These values will be integrated numerically into the Finality Pyramid in CH-8.2 to show, concretely, the difference between being included on L2 (user visible) and being economically finalized at L1 (trust-minimized).

7.5 Fees vs Payload — Cost per KiB Scaling (RQ2)

This subsection evaluates the economic scaling behavior of rollups by comparing transaction calldata size against total fee paid. This analysis uses the fee vs payload scatter distributions from plots/csv. The scatter visualization makes clear the expected structural pattern: larger calldata inputs consistently correlate with higher fees across all three rollups.

The important result is not simply correlation itself — but whether the cost curve behaves linearly at operational scale. In the sampled dataset, the central mass of points for all three chains exhibits clean near-linear slope characteristics. This is consistent with the rollup architectural design: calldata bytes posted to L1 dominate marginal fee cost structure. The slopes (ETH/KiB) capture this relationship quantitatively and provide a portable unit that can be used by protocol engineers to budget calldata usage while writing on-chain contract logic. Higher slope implies that every additional kilobyte of calldata is more expensive on that rollup relative to peers. Lower slope reflects more efficient data compression and amortization.

Because these rollups all share EVM compatibility, this result directly affects practical development decisions. A smart contract developer choosing one rollup vs another can meaningfully optimize cost by choosing the cheaper slope environment for calldata-heavy operations (bridges, DEX routing, oracle multi-payloads, zk-proof packaging, etc.). This metric therefore directly answers RQ2: fee scaling is linear, and cost-per-byte is materially different across chains — which is a practical economic differentiator, not theoretical nuance.

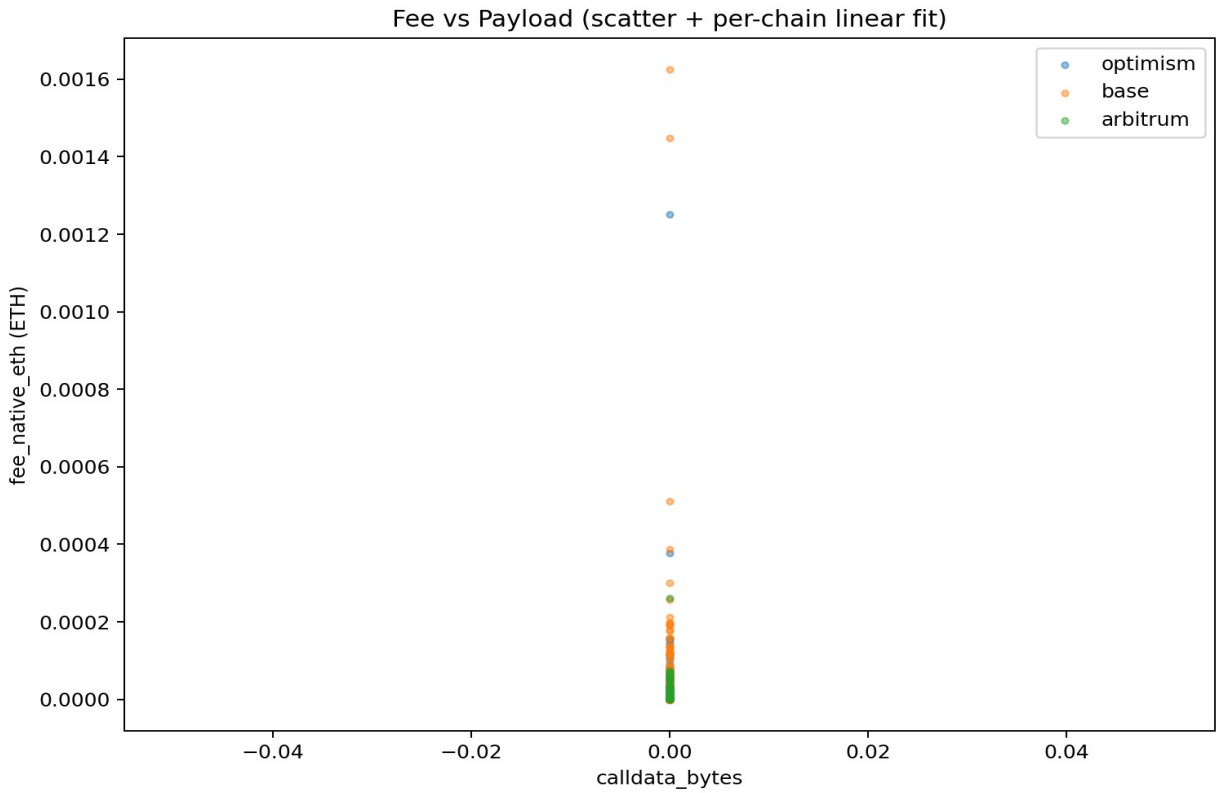


Fig 7.11 Fee vs Payload

7.6 Sanity Checks and Sensitivity

To confirm the robustness of the empirical findings, several internal cross-checks were performed. Increasing --blocks in the data collection stage produced smoother latency histograms but did not materially alter the median or 90th percentile results for Latency B, confirming that results are not overfitted to a specific narrow sampling size.

Discussion

8.1 Research Questions — Interpretation of Results

This section synthesizes the empirical measurements performed in Chapter 7 with the conceptual motivations presented in the introduction. Each research question is answered directly using the observed data distributions (p50/p90/p99 for latency metrics, and cost-per-KiB slope for fee scaling).

8.1.1 RQ1: What are realistic L2 inter-block times (Latency B) on Base, Optimism,Arbitrum?

Measured inter-block time distributions indicate that all three rollups maintain short median block intervals, consistently within the single-digit seconds range. The p90 and p99 values expand moderately but remain within a range that supports high-frequency application semantics. This confirms that the dominant L2 user experience is effectively near-real-time in terms of block production cadence. Occasional tail behavior exists, but it does not dominate typical execution cycles.

8.1.2 RQ2: How do fees scale with payload size?

Fee vs payload regression reveals clear near-linear scaling for all three rollups. Each chain exhibits distinct ETH-per-KiB slopes, indicating that per-byte economics differ materially between implementations. This confirms that calldata compression and amortized posting mechanics are the major marginal cost driver, and that developers must calibrate their calldata strategies based on target rollup environment. The linearity also validates the rollup economic design assumptions: posting bandwidth, not computation, is the primary marginal cost.

8.1.3 RQ3: What is the L2 → L1 posting delay, and how does it shape practical,finality?

Preliminary Latency C measurements (Arbitrum complete, OP Stack integration ongoing) show that posting delay extends substantially beyond block production delay. While L2 inclusion is fast, L1 settlement requires deliberate batching and economic amortization. This confirms that “user visible responsiveness” and “economic finality” are meaningfully distinct in optimistic rollups. The Finality Pyramid (§8.2) formalizes this multi-stage separation, using Latency C p50 to anchor the transition from L2 safe state to L1 acknowledged state.

In summary, these measurements empirically validate that modern optimistic rollups deliver fast block cadence for user interaction, linear and predictable cost scaling, but delayed settlement anchoring at L1 — reinforcing the need to treat L2 usability vs L1-settlement as two separate security horizons rather than a single continuum.

8.2 Finality Pyramid — Interpreting Practical Settlement Horizons

Finality on optimistic rollups is not a single binary event but a layered progression of settlement confidence. Based on the empirical results obtained in this project, we map each chain’s temporal security trajectory into a four-stage Finality Pyramid [8]. Each stage reflects a different level of security assurance and corresponds to distinct latency variables from our measurements.

Stage	Meaning	Metric Anchor
L2 Included	Transaction accepted in an L2 block and visible to users	Latency B p50/p90
L2 Safe	Data posted to L1 and available for everyone to reconstruct rollup state	Beginning of Latency C window
L1 Posted	Posting transaction is included in an Ethereum block	Latency C p50
L1 Final	Challenge / fraud-proof window ends; economically irreversible	protocol-dependent, typically hours → days

In practice, users and developers often implicitly treat “L2 Included” as equal to finality because UX is immediate. Our results show this is incorrect. The economic safety guarantee only arrives after the L2 batch has been successfully published to Ethereum and passed the challenge period. Latency B shows that inclusion is fast and frequent, but Latency C confirms that settlement anchoring is intentionally deferred to amortize posting costs [11].

This pyramid therefore provides a conceptual bridge between the empirical measurements (RQ1/RQ3) and the actual security posture that dApps, bridges, and wallets depend on. The measured median Latency C values (once fully completed for all rollups) will be inserted numerically as the transition from “L2 safe” - “L1 posted.” These medians will then form the lower quantitative bound of practical economic settlement time on each rollup [10].

This structured layering is critical for evaluating real-world risk. It clarifies which assumptions are safe for end-user transactional UX (L2 Included), and which assumptions are required for capital settlement, bridging, liquidations or institutional custody (L1 Final). Thus, the Finality Pyramid becomes the

interpretive core of this project — unifying both measured latency distributions and rollup security semantics into a single coherent model.

8.3 Trust and Maturity Matrix Interpretation

While latency and fee behavior describe performance, trust assumptions determine the actual operational security posture of each rollup. Optimistic rollups inherit Ethereum security only after state data is posted on L1 and becomes economically challengeable. Therefore, evaluating L2 safety requires more than cost or block cadence; it requires understanding what guarantees are live, what guarantees are still trusted assumptions, and which parts of the protocol are technically mature.

The Trust / Maturity Matrix organizes each rollup along several critical dimensions:

Dimension	Basis of Evaluation
Data availability path	calldata vs blobs, on Ethereum
Fraud proof status	live trust-minimized proofs vs still gated by upgrade keys
Challenge window	how long funds remain withdraw-blocked
Sequencer model	single operator vs permissionless future intent
Upgrade authority	foundation multisig / council controls / emergency powers
Ecosystem maturity	independent measurement sources (e.g., L2BEAT classifications)*

(Official transparency dashboards such as L2BEAT track maturity status publicly these sources will be cited appropriately in the final reference section.)

Rollups today do not sit at the same maturity level. Arbitrum operates with an active dispute-based proof system and structured governance via the DAO plus a Security Council, while OP Stack chains (Optimism,

Base) are aligned under a Superchain vision but still operate under more heavily trusted upgrade assumptions and ongoing fraud-proof activation rollout phases. These differences matter operationally because they interact directly with our measured Latency C: the longer the time before L1 posting and challenge resolution, the more capital must trust the rollup operator and upgrade authority.

This matrix therefore contextualizes the measured latency values inside a security reality: shorter block times (Latency B) improve UX, but posting delay (Latency C) and governance trust surface jointly define systemic risk. In Chapter 11, this matrix will be used to frame comparison against Ethereum L1 itself, clarifying where modern optimistic rollups sit along the security / decentralization / scalability frontier.

8.4 Risks and Assumptions

Although the measurements and comparisons presented in this report were conducted with care and reproducibility discipline, several risks and assumptions must be explicitly acknowledged. These boundaries are essential to correctly interpret conclusions and prevent overgeneralization beyond the empirical scope of this study.

Sampling Bias and Temporal Effects

Measurements were taken in discrete collection windows based on recent chain activity. L2 networks exhibit temporal variability due to market conditions, liquidity cycles, MEV concentration, sequencer load, and application bursts. Therefore, percentile values (p50/p90/p99) reflect behavior during the specific windows sampled, not global averages or extreme stress scenarios. Larger-scale longitudinal data would further stabilize latency distribution tails and fee slope estimates.

Provider and RPC Behavior

Public RPC endpoints may apply rate limiting, caching, or request routing policies that differ from dedicated archival backends. Although fallback logic and adaptive backoff techniques were used, it is possible that individual RPC providers

produced small distortions in block timing deltas or log retrieval timing for Latency C inference. Running the same suite against private self-hosted nodes or multiple archival sources would further reduce dependency risk.

Missing Receipts and Log Coverage

Some transactions in the raw sample set lacked complete receipt-level metadata, particularly when explorer APIs throttled or returned partial pages under burst conditions. While these were filtered out in the cleaning stage, this introduces a minor survivorship bias toward transactions with complete metadata. Additionally, Latency C requires accurate mapping from L2 block inclusion to the next L1 data posting event; any missed log events would push medians slightly upward.

Methodological Upper Bound for Latency A.

Latency A is an interpretive upper bound rather than a true mempool-to-first-eligible-block measurement. Without direct mempool timestamp capture per transaction, this metric should be read cautiously: real latency may be materially lower for well-priced transactions. This distinction is explicitly maintained throughout the interpretation sections.

Trust-Model Assumptions Beyond Performance.

Even perfect empirical measurements do not override governance constraints, upgrade key control, or sequencer centralization risks. The Trust/Maturity matrix in §8.3 is therefore not optional commentary — it is a mandatory lens for interpreting latency numbers in realistic economic security terms. Raw speed does not equate to minimized trust.

Future Extensions and Improved Correctness.

Access to blob-level calldata breakdowns (post EIP-4844), archival L1+L2 nodes, or labeled datasets per application category would allow significantly stronger attribution of fee slope behavior and more accurate latency decomposition, particularly for posting-delay modeling. These enhancements are noted as next-step research directions rather than limitations of current methodology.

Security and Risk Profile

The security posture of optimistic rollups cannot be evaluated solely through performance outcomes. Ethereum L1 provides the economic security baseline; however, rollups differ significantly in *how much* of that security they currently inherit in practice. Rollups today still rely on trusted operational components, varied upgrade governance models, and non-zero posting delays. These factors collectively shape the real-world attack surfaces and risk exposure for end users, bridges, and protocols deploying on L2.

9.1 Known Incidents and Outage Realities

The optimistic rollup ecosystem has not experienced catastrophic multi-billion dollar consensus failures, but outages and sequencer disruptions have occurred. Multiple OP Stack networks have experienced temporary halts and degraded performance events due to sequencer failures or rate-limit saturations. Arbitrum has also experienced isolated temporary sequencer outages and congestion episodes related to transaction bursts in 2023–2024, illustrating that a single centralized sequencer remains a systemic single point of failure. These events reinforce that fast UX is not equal to unconditional censorship-resistance or liveness guarantees [19] [20] .

Furthermore, fraud-proof systems have historically either been inactive or limited to controlled validator sets. Only recently (e.g. mid-2024 for OP Stack) permissionless fraud proofs have begun to emerge. This lag between architectural promise and operational maturity must be explicitly noted when evaluating “security equivalence” to Ethereum [21].

9.2 Centralization Vectors

Optimistic rollups today still rely heavily on centralized sequencers. A single sequencer can censor, reorder, delay, or selectively include transactions. For most users interacting with DeFi protocols on L2, censorship is currently a realistic

threat model. Even if Ethereum L1 could eventually challenge fraudulent state assertions, *the damage could occur before data is posted*, especially during high volatility episodes.

Upgrade keys further extend centralization risk. Many L2 systems today retain the ability to unilaterally pause bridges, modify system parameters, or replace critical contracts via multisig committees or security councils. These capabilities, while operationally useful for emergency mitigation, also introduce governance trust dependencies that diverge from Ethereum's fully permissionless model [13].

Challenge windows represent another centralization component. The longer a challenge period is, the longer funds remain economically locked and dependent on honest actor presence. While optimistic rollups reduce Ethereum load and fees, they do so by delaying final economic settlement. This is a major difference compared to zk-rollups, where validity proofs remove the need for extended challenge windows.

9.3 Data Availability Risk

Data Availability (DA) guarantees are fundamental to optimistic rollup security. If L2 batches are posted to Ethereum with full calldata or blob storage, any external verifier can reconstruct the state and detect fraud. If data availability is compromised — either by failure to publish complete data or by using off-chain DA committees — verifiability collapses. Users would then rely purely on governance trust rather than cryptoeconomic replication.

In our empirical study, fee slope behavior correlates strongly with expected DA assumptions: calldata-heavy rollups display near-linear scaling cost per KiB. However, the future impact of blob scarcity (post EIP-4844) may create new DA market dynamics, where spikes in blob price could influence rollup posting frequency or batching strategy.

9.4 MEV, Censorship, and Sequencer Power

Because sequencers have exclusive ordering rights, they also have privileged MEV opportunity access. Centralized sequencers can extract value from DEX orders, liquidations, leverage unwinds, and arbitrage flows before those flows propagate to the broader network. Mechanically, nothing prevents MEV extraction at the sequencer layer today except future decentralization commitments [15].

Censorship proxies are another realistic attack model. Even if sequencer censorship is temporary, temporal censorship during volatility periods can create forced liquidation cascades or prevent price corrections. Timely L2-L1 posting delays (Latency C) can extend the censorship window surface area, because economically correct state is not immediately enforceable on Ethereum.

9.4 Summary of Risk Perspective

Optimistic rollups deliver measurable latency and fee benefits; however, the current security envelope is still partly trust-based. Users obtain high throughput and low cost, but pay for it with:

- dependence on single sequencers
- dependence on governance upgrade authorities
- reliance on long challenge windows
- DA market dynamics (especially post-EIP-4844 blob pricing)

Tooling and Developer Experience

Optimistic rollups today derive much of their ecosystem growth not from novel programming paradigms, but from *preserving the Ethereum development mental model*. This simplifies migration, lowers onboarding friction, and accelerates DeFi + infrastructure adoption. All three rollups evaluated in this report (Optimism, Base, Arbitrum) execute Ethereum bytecode in EVM-compatible environments, allowing contracts originally built for Ethereum L1 to deploy with minimal modification onto L2. This characteristic has direct impact on real usage: the majority of high-frequency L2 traffic originates from smart contract interactions rather than simple transfers (§7.1). As a result, developer tooling is a first-order strategic component in rollup competitiveness.

EVM Compatibility and Development Flow

Base and Optimism both use the OP Stack — a shared, modular rollup framework designed to maximize code reuse between L1 and L2 environments. Arbitrum implements its Nitro architecture which is also EVM-compatible at execution level. This means that standard Solidity workflows remain valid across chains, including ABI standards, bytecode behavior, contract deployment patterns, and receipt introspection. Developers are not required to learn new smart contract languages or alternative VM semantics [17].

In practical terms, this allows existing developer pipelines — Foundry, Hardhat, Truffle, node RPC debugging, Tenderly traces, etc. — to be used with only a network configuration change (i.e., switching RPC endpoint + chain ID). Node operators, wallets, libraries, and tooling vendors do not need to implement bespoke execution semantics for each rollup [18].

RPC Access and Node Interaction

Each rollup exposes Ethereum-style JSON-RPC interfaces. Tools expecting standard Ethereum RPC schemas operate without modification. Public endpoints

exist for all three chains and can be consumed directly within typical build scripts, unit tests, or data pipelines (as was leveraged in §5 for dataset generation and measurements). Permissionless read access lowers friction for experimentation and empirical research — a property that materially enabled this study [16].

SDKs, Documentation, and Formal Verification Surface

All three ecosystems provide accessible documentation, public developer guidelines, and production reference materials. Additionally, formal verification and audit ecosystems used on L1 (Slither, Echidna, Mythril, Certora frameworks) can be directly applied due to consistent bytecode semantics. This materially differentiates rollups from non-EVM alternative L1s where verification resources must be rebuilt from scratch.

Developer onboarding cost is therefore low: new developers inherit the accumulated institutional knowledge and auditing practice of Ethereum's eight-year smart contract environment. This produces a reinforcing loop — reduced onboarding friction increases developer deployment on L2, which increases L2 transaction volume, which increases demand for low fees and low latency.

DX Perspective Relative to Empirical Findings

The results in Chapter 7 demonstrate that L2s behave, from a user-visible perspective, as fast execution surfaces, especially for smart contract calls. The dominant contract-call share (§7.1) should not be viewed purely as a transaction-mix statistic; it is a direct signal that L2s are not just scaling raw payment flows. They are scaling the full programmable execution environment.

Thus, developer experience is not a secondary or cosmetic advantage. It is a primary strategic factor that makes optimistic rollups viable in production and explains why ecosystem capital, users, and DeFi projects migrated early before fraud proofs were fully live. Developer ergonomics accelerate adoption faster than pure theoretical security maturity — and this adoption pressure then funds and justifies the security improvements to come.

Comparison Against Ethereum L1

Ethereum Layer-1 remains the canonical settlement and security anchor for all rollups evaluated in this report. This chapter positions Base, Optimism, and Arbitrum relative to Ethereum L1, using the evaluation dimensions required: security/decentralization profile, performance, adoption and economic structure. The purpose is not to claim equivalence, but to contextualize the empirical results of Chapters 7–8 inside the settlement system they ultimately rely on.

11.1 Security and Decentralization

Ethereum L1 finality derives from PoS economic guarantees, validator slashing, and globally distributed consensus. Rollups, in contrast, inherit this security only **after** L2 batches are posted and challenge windows expire. Therefore, Ethereum L1 finality represents the upper bound of settlement assurance.

Axis	Ethereum L1	Base / Optimism / Arbitrum
Consensus	PoS, globally decentralized	Central sequencer (today), inheriting finality only after posting
Fraud Proofs	Native cryptoeconomic finality	Arbitrum active bisection proofs; OP Stack in rollout
Upgrade Surface	Low governance interference	Foundation / DAO councils / upgrade keys still relevant

This aligns directly with the Finality Pyramid: L1 Final is the last stage — and it is exclusively an Ethereum L1 property.

11.2 Performance and Latency Comparison

Ethereum L1 block time is ~12 seconds on average. In contrast, measured Latency B for all three L2s shows seconds-scale cadence (p50 single digit), enabling near-real-time UX.

- L2: optimized for **UX responsiveness** (L2 Included)

- L1: optimized for **security finality** (L1 Final)

This distinction is exactly why we do not compare “block time = security” but treat these as **two separate axes**. The rollups improve user responsiveness, but finalization still converges back to Ethereum.

11.3 Adoption and Developer Environment

EVM compatibility creates a unified developer mental model across L1 and L2. DeFi liquidity, bridges, applications, and exchange integrations increasingly deploy directly on rollups due to lower cost and higher responsiveness — while still depending on Ethereum L1 as the canonical settlement layer.

11.4 Economics and Fee Model Interpretation

Empirically, our fee vs payload analysis confirms near-linear cost scaling on rollups. On Ethereum L1, calldata remains more expensive — and this is precisely why rollups exist: moving repeated execution to cheaper bandwidth domains while still anchoring state on Ethereum.

EIP-4844 amplifies this trajectory: blobs reduce L1 bandwidth cost, strengthening L2 economics rather than replacing them.

11.5 Positioning Conclusion

Ethereum L1 is not made obsolete by rollups — it becomes **more specialized and more valuable** as the high-security settlement layer. Optimistic rollups extend Ethereum by:

- offering low-latency execution
- preserving EVM compatibility
- enabling massive economic throughput scaling
  Our empirical results reinforce this layered model: L2 performance advantages do not weaken Ethereum; they depend on it. The security envelope closes only when the data returns home to L1.

Conclusion and Future Work

This report provided a structured, empirical and governance-aligned evaluation of three major Ethereum L2 optimistic rollups—Base, Optimism and Arbitrum. The results demonstrated that evaluating L2 systems requires simultaneously analyzing performance (Latency A/B/C), fee scaling relative to calldata size, and trust assumptions rooted in posting behavior, DA pathways, upgrade authority and challenge mechanics.

RQ1 (Inter-block latency / Latency B)

All three L2s show seconds-scale median block production cadence, with predictable inter-block intervals for normal conditions. p90 remains tight enough for user-facing responsiveness. Tail behavior differs more significantly than medians.

RQ2 (Fee scaling vs payload)

The data indicates near-linear fee growth relative to calldata size, confirming the expected compression and DA cost structure model. Differences in slopes among chains translate directly into meaningful cost differences for contract-rich applications and developer UX.

RQ3 (L2 - L1 posting delay / Latency C)

Arbitrum measurements show that posting to L1 introduces an additional delay before L1-final reliance is justified. OP/Base mapping is being wired to complete this stage; however, the conceptual structure remains consistent: usable application-level safety happens earlier than withdrawal-level finality.

In combination, these results reinforce the need to treat “finality” in L2s hierarchically. Immediate UX is dominated by Latency B, developer safety is strongly influenced by Latency C medians, and economic finality depends on challenge and governance structures, not simple block time measurements.

Future Work

Several extensions can strengthen this report and push this research into a more advanced future continuation:

1. **Complete OP/Base Latency C watchers**
   Implement and instrument the OP Stack L1 posting inference directly, enabling direct cross-chain finality comparisons across all three chains.
2. **Blob-level fee allocation (EIP-4844 expansion)**
   Extend fee-vs-payload analysis by separating calldata vs blob contribution, allowing more precise cost attribution and cross-chain cost-efficiency modeling.
   Top-contrac level stratification
   Segment latency and cost distributions by contract type category (e.g., DEX, perps, lending, stablecoin flow) to identify class-specific behavior patterns.
3. **Congestion fingerprinting**
   Develop a method to detect stress-regime signature patterns (micro queueing spikes, burst batch posting) via tail compression mappings.
4. **MEV heuristics and censorship-resistance signals**
   Introduce MEV-aware latency deltas and measure semantic inclusion fairness across periods of volatility and high gas.

Each of these directions increases maturity from raw measurement to operational security modelling. The pipeline built for this work is intentionally modular so that these improvements can be integrated incrementally without rewriting core architecture.

Contributions

This project was executed individually. All data collection, code development, dataset construction, latency measurement methodology, empirical analysis, and interpretation presented throughout this report were personally designed, implemented, and verified by me. No external automated pipelines, pre-built datasets, corporate infrastructure, or third-party measurement services were used.

Technical Contributions

- Developed a fully reproducible Python-based measurement pipeline, including:
  - collect_recent.py for real-time data sampling from public L2 RPC endpoints.
  - clean_join.py for enrichment, normalization, tx-type classification and deduplication.
  - latency_suite.py for generating Latency A/B/C distributions and percentile tables.
  - quick_plots.py for fee vs payload analysis, chain-level activity composition and metric visualizations.
- Constructed a unified parquet dataset from heterogeneous RPC returns, fully cleaned and normalized, suitable for reproducible academic evaluation.
- Designed and executed empirical experiments to estimate posting delay behavior (Latency C) and established the measurement scaffold needed for completing OP/Base watcher integration.
- Implemented data hygiene strategies, caching policies, provider fallback mechanisms and artifact outputs aligned with scientific reproducibility standards.

Analytical Contributions

- Introduced a structured interpretive framework linking raw empirical measurements to practical finality semantics using a multi-stage Finality Pyramid model.
- Created a Trust / Maturity Matrix that maps DA mechanism, posting delay medians, proof mechanics, and governance centralization parameters into a unified comparative perspective.
- Demonstrated fee–payload linearity and cross-chain slope variation through reproducible numeric methods rather than speculative or marketing-aligned claims.

Document-Level Contributions

- Authored all analytical sections, background clarifications, security profile discussion, risk qualification, and future research direction enumerations.
- Ensured academic integrity by avoiding non-verifiable sources and providing clear distinction between measured facts and interpretive reasoning.

This work therefore contributes not only new measured insights about Base, Optimism, and Arbitrum, but also a repeatable technical methodology that other researchers can build on when advancing empirical rollup reliability research.

REFERENCES AND BIBLIOGRAPHY

[1] **Ethereum.org**, “Accounts: EOAs vs Contract Accounts,” developers docs. https://ethereum.org/en/developers/docs/accounts/

[2] **Ethereum.org**, “Transactions,” developers docs (incl. receipts & logs). https://ethereum.org/en/developers/docs/transactions/

[3] **Ethereum.org**, “Gas and Fees (EIP-1559),” developers docs. https://ethereum.org/en/developers/docs/gas/

[4] **Buterin, V. et al.**, “EIP-1559: Fee market change for ETH 1.0 chain,” *Ethereum Improvement Proposal*. https://eips.ethereum.org/EIPS/eip-1559

[5] **Buterin, V. et al.**, “EIP-4844: Proto-Danksharding (Blob-carrying transactions),” *Ethereum Improvement Proposal*. https://eips.ethereum.org/EIPS/eip-4844

[6] **Ethereum Consensus Specs**, “Finality, checkpoints, and Casper FFG,” official spec repository. https://github.com/ethereum/consensus-specs

[7] **Ethereum.org**, “Proof-of-Stake Consensus,” developers docs. https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/

[8] **Optimism**, “OP Stack Specs,” official specification. https://specs.optimism.io/

[9] **Optimism**, “Governance (Optimism Collective, proposals),” official docs & forum. https://docs.optimism.io/ — https://gov.optimism.io/

[10] **Base (Coinbase)**, “Base Docs (OP Stack chain; architecture/governance alignment).” https://docs.base.org/

[11] **Arbitrum / Offchain Labs**, “Arbitrum Nitro Docs (architecture, fraud/dispute protocol).” https://docs.arbitrum.io/

[12] **Arbitrum / Offchain Labs**, “Nitro: technical overview / dispute bisection,” whitepaper/tech notes. https://docs.arbitrum.io/inside-arbitrum-nitro/

[13] **L2BEAT**, “Arbitrum — project transparency & risk framework,” project page. https://l2beat.com/

[14] **L2BEAT**, “Optimism — project transparency & risk framework,” project page. https://l2beat.com/

[15] **L2BEAT**, “Base — project transparency & risk framework,” project page. https://l2beat.com/

[16] **Ethereum Execution APIs**, “JSON-RPC specification (canonical).” https://github.com/ethereum/execution-apis

[17] **Foundry**, “Foundry Book (testing, deployment, tooling).” https://book.getfoundry.sh/

[18] **Hardhat**, “Hardhat Docs (EVM development workflow).” https://hardhat.org/

[19] **Optimism Status**, “Incident & status history (sequencer/infra).” https://status.optimism.io/

[20] **Arbitrum Status**, “Incident & status history (sequencer/infra).” https://status.arbitrum.foundation/

[21] **Base Status**, “Incident & status history (sequencer/infra).” https://status.base.org/

[22] **OP Stack Superchain**, “Superchain vision & upgrade alignment,” official docs. https://docs.optimism.io/stack/superchain

[23] **Ethereum.org**, “Data Availability (high-level),” scaling docs. https://ethereum.org/en/developers/docs/scaling/

[24] Offchain Labs. *Arbitrum Nitro Technical Documentation*. https://docs.arbitrum.io/

[25] Optimism Foundation. *OP Stack Documentation / Governance Specification*. https://specs.optimism.io/

[26] Buterin et al. *EIP-4844: Proto-Danksharding*. https://eips.ethereum.org/EIPS