

Paper:

[https://drive.google.com/file/d/1zY4b2XjkN6ih9\\_-QdiP66YAi-1QEBJVj/view?usp=sharing](https://drive.google.com/file/d/1zY4b2XjkN6ih9_-QdiP66YAi-1QEBJVj/view?usp=sharing)

Source code:

[https://github.com/souradeep-das/finality\\_index](https://github.com/souradeep-das/finality_index)

# Reliablocks: Developing Reliability Scores for non-finalized blocks in Optimistic Rollups - Devcon Hacker House 2024

Souradeep Das, Ethan Lam, Varun Vaidya, Sanjay Amirthraj

## Abstract

Introducing Reliablocks, an on-chain reliability index for non-finalized blocks in Optimistic Rollups. This was built during the EigenLayer Infinite Hackathon at the Infinite Hacker House at DevCon 2024. As part of this research, we delivered a working Layer AVS WASM component, a working Eigen Layer AVS component, a working EigenLayer Solidity smart contracts that work with the AVS component, and a UI dashboard illustrating the reliability score and a derived interest rate for further utilization.

## Introduction

Optimistic rollups are Layer 2 scaling solutions for Ethereum that process transactions off-chain and post results on-chain, offering improved scalability and approximately 99% lower fees, while inheriting Ethereum's security. These solutions have gained significant traction, with the top two projects—Arbitrum and Optimism—currently holding \$21.2 billion in assets.

Fast exit providers such as Celer and Across play a crucial role in this ecosystem by fronting funds to users, allowing them to receive funds quicker than Optimism's standard 7-day finalization period when bridging funds to the Ethereum mainnet. This service comes with an interest charge, compensating providers for the risk they assume.

However, if you are not running a validator, you have no way of determining whether a specific block will be finalized or not. Moreover, for the general Layer 2 (L2) user, it is neither feasible nor expected to run a validator. Thus, they have no reliable way to identify whether a transaction they were a part of was valid (inherited validity if it is on top of a valid state) and will not be rolled back during the next seven days. This raises an important question: How long should a user on an L2 wait before they can deem their transaction final?

## Reliablocks

We are building an on-chain reliability index for non-finalized blocks in Optimistic Rollups to help non-validators estimate the likelihood of a non-finalized block being finalized. Such an index would provide valuable information to users without requiring them to trust a specific entity or validator for the information. A dynamic score, instead of a binary final/non-final answer, also assigns a weight to the reliability of what multiple agents think about the finality of a specific block.

## AVS Integration for Reliability Score

One of our (initial) key metrics for determining the reliability score is fast exits. AVS (Actively Validated Services) makes it possible to collect and translate fast exit data into a reliability score.

Every Layer 2 (L2) transaction is batched and deposited onto Layer 1 (L1), which means that every fast exit transaction data is available on L1 and can be easily verified on-chain. However, since it is part of the calldata, it is not retrievable by an external contract but is still verifiable on-chain. Additionally, the counterpart of the fast exit on L2, which is a liquidity transfer on L1, is also available on L1. While both of these are verifiable on-chain, they need to be inputted by an operator. This is where AVS comes into play, helping to set up a validation mechanism where operators provide data and are slashed for providing fraudulent information.

Furthermore, rather than simply providing a binary final/non-final answer for each L2 block, the validator assigns a numerical value, quantifying the finality they are proposing. This approach creates an effective way for end users to trust multiple operators at once by referring to a single metric, while also giving the conditions of trust to the users' discretion.

Overall, AVS supports this research project by offering several key features:

- Slashing a validator who provides incorrect data, utilizing restaked security guarantees.

- Making the reliability index available on-chain, so it can be further utilized by other cross-chain bridges, smart contract logic, and more. (For instance, users can be descriptive with their risk indices when interacting with a contract.)
- The on-chain reliability index can also be deposited to L2, where it will relay the index data back to L2 decentralized applications (DApps), extending its application to apps running on L2.

See Figure 2 for a visual image of fast exit reliability metric determination.

An on-chain reliability index for non-finalized blocks in Optimistic Rollups would greatly enhance user experience by reducing uncertainty and improving risk assessment for transactions. It would provide valuable insights for users and fast exit providers, enabling informed decisions regarding withdrawals and fund fronting. This increased transparency could boost confidence in Optimistic Rollups, facilitating better interoperability solutions and greater adoption of Layer 2 solutions on Ethereum.

## Introducing Our Score

### Quantifying L2 Block Finality

How can an L2 block's finality be quantified? Every L2 has certain actors who actively note down or make decisions based on the validity of a state, even to the extent of putting money at risk to support their decisions. These actors are assumed to run a full validator. Who are they? They include fast exit providers and liquidity pool providers who take someone's position on the L2 themselves, ensuring the validity of the rollup's state while providing fast liquidity in return.

### Dynamic Reliability Score

Our score is a metric to determine the likelihood of block validity, using the number and value of fast exits in an L2 block as a key determinant. Blocks with more fast exits, especially high-value ones, are considered more reliable, as they indicate validators' confidence in the block's validity and confirm the changed state of the block. Since each block is assigned a financial weight based on the fast-exit providers enabling exits on it, our score becomes dynamic. Every subsequent fast-exit provider effectively attests to the validity of the previous state on which they are providing the service. Over time, a block's reliability score increases, akin to block confirmations on Layer 1 (L1). These cumulative scores serve as a suitable indicator for entities, such as exchanges, to wait for a specific reliability score before accepting a transaction on an L2.

### Integration into Cross-Chain Products

To integrate the score into cross-chain products, we use it to determine interest rates. Blocks with higher reliability scores are associated with lower interest rates, reflecting lower perceived risk. We model this relationship using a negative exponential decay function to assign a score between 0 and 100, normalized for each block. Based on this normalization, interest rates are expected to range between 1% and 3%, depending on the score.

## Architecture

See Figure 1

## Primary Stakeholders

The primary stakeholders in this system include:

1. Optimistic rollup users who want to ascertain the finality of their transactions.
2. Optimistic rollup users seeking faster withdrawals.
3. Centralized exchanges (CEXs) on L2 requiring a reliable indicator (confirmations) to accept certain deposit transactions.
4. Fast exit providers who validate transactions and assume financial risk.
5. Rollup operators and validators responsible for processing transactions.
6. The proposed reliability scoring system, which could evolve into a fast exit provider itself.

## Future Steps

We plan to develop stronger indicators that provide a clearer understanding of whether a block has finalized. Additionally, we aim to create an insurance protocol to secure blocks further. Blindly trusting finality in rollup bridges often requires trust in a set of validators. What if there was a way to determine how trustworthy these validators are? Our proposed scoring system addresses this concern by offering a transparent and dynamic measure of reliability, reducing the need for blind trust

in specific entities.

## **Hackathon Development and Feedback: 11/10-11/15**

### **1. Setup (11/10-11/11)**

- Learned to build on Layer (new platform for us).
- Successfully ran the square and oracle example.

Walking through the Layer docs was easy — we were able to obtain funds from the faucet, deploy contracts, and run the tests for the square and oracle examples.

### **2. Challenges with Rust Smart Contracts on Layer's Chain**

#### **2.1 Fetching Issues**

- Unable to interact with smart contracts on Layer.
- Difficulties in querying contracts.

#### **2.2 WASI Component Setup**

- Successfully set up WASI components.
- Implemented result returning and wasmatic test components.

We built the off-chain WASI component that would allow operators to submit tracked fast exit transactions from Optimism. The WASM component takes a Layer 2 (L2) block number as input and returns the reliability score as a response. We were able to run tests for these WASI components we developed. The component was deployed to the Task Queue contract, and the wasmatic run/test commands were executed successfully to test the request responses to our WASM.

### **3. Encountered Layer-Specific Challenges (11/14-11/15)**

#### **3.1 Task Queue Problems**

- Empty task queue when attempting to read records/tasks.
- Successful CLI task running and testing, but unclear on-chain response confirmation.

#### **3.2 Documentation Gaps**

- Lack of advanced contract CLI command documentation.
- Insufficient information on actual contract deployment interaction.
- Limited resources on Layer SDK and underlying Rust blockchain.

### **4. Contract Interaction Hurdles**

#### **4.1 Contract Querying**

- Lack of clarity on the "msg" option format in CLI for contract queries.

#### **4.2 Storage Layout**

- Unclear contract storage layout, especially for default and result storage.

### **5. Alternative AVS Development**

- Successfully built an alternate AVS using Layer.
- Had trouble connecting to the frontend.

### **6. Parallel Development on EigenLayer**

- Built AVS on EigenLayer as well.

- Achieved a working iteration in 3-4 hours (compared to 3 days on Layer).

## 7. Identified Areas for Improvement

- More comprehensive documentation needed for Layer.
- Better resources required for Cosmos and Rust blockchain interactions.
- Need for more robust contract interaction tools and documentation.

## Citations

1. L2Beat. "TVL in Optimistic Rollups". Available at: <https://l2beat.com/scaling/tvl>
2. Aft 2023. "LIPIcs: AFT.2023.22". Available at: <https://drops.dagstuhl.de/storage/00lipics/lipics-vol282-aft2023/LIPIcs.AFT.2023.22/LIPIcs.AFT.2023.22.pdf>
3. Ethereum.org. "Optimistic Rollups". Available at: <https://ethereum.org/en/developers/docs/scaling/optimistic-rollups/>
4. Alchemy. "Overview of Optimistic Rollups". Available at: <https://www.alchemy.com/overviews/optimistic-rollups>
5. OSIZ Technologies. "Optimistic Rollups Development". Available at: <https://www.osiztechnologies.com/blog/optimistic-rollups-development>

## Figures

Figure 1: Hand-drawn diagram of ReliaBlocks architecture

[

Hand-drawn diagram of ReliaBlocks architecture

820×1080 79.5 KB

](<https://canada1.discourse-cdn.com/flex028/uploads/eigenlayer/original/2X/d/dfc7a82a2b6c18a1fd7003298f948194dc692ff7.png>)

Figure 2: Visual Description of Fast Exit Reliability Definition

[

Visual Description of Fast Exit Reliability Definition

1196×962 78.7 KB

](<https://canada1.discourse-cdn.com/flex028/uploads/eigenlayer/original/2X/7/7619a323c3decead778ecaa4a0da65c458191eec.jpeg>)