MACH for Faster Finality

To address the slow finality of rollups, we present MACH: a fast finality layer for Ethereum rollups with the following key desiderata: 1. 1. 2. Fast confirmation for rollup transactions 3. 2. 4. Crypto-economic security so as to handle any malicious network participants 5. 3. 6. Support both ZK as well as optimistic rollups 7. 4. 8. Generic enough to support different proof systems and runtimes In order to guarantee finality, MACH as a network needs to verify the validity of a rollup state to ensure that the rollup operators have followed the state transition function correctly. To this end, MACH supports three state validity modes. Pessimistic Mode

In the pessimistic mode, each transaction is considered by default invalid, and hence needs to be replayed. As a result, the rollup operator feeds transaction data directly to the MACH network which in turn re-executes the transaction and reaches consensus on the validity of the proposed state by the rollup operator. While this operational mode is the simplest of all, one of its major downsides is that it is not very efficient as MACH practically operates as a network of full nodes for the rollup. And this results in beefy node requirements. A future work is to build stateless clients that require a smaller state footprint to operate a node for the rollup. Optimistic Mode In this mode, the rollup operator asserts a state claim on MACH, stating that the execution of a specific block of transactions leads to a particular state commitment. Any node in the MACH network can then challenge the claim and prove that the new state is not valid by engaging with the rollup operator in a bisection protocol. This is the classical Optimistic Mode as shown in the diagram below.

Note that the bisection protocol is triggered only when a challenger believes the state commitment is not valid. Alternatively, one could also replace the bisection protocol by on-demand ZK proofs, where the ZK proof is only generated if there is a challenge. This is explained in the diagram below.

This setup assumes the existence of at least one honest node in the MACH network and network nodes are mostly in observation mode. Validity Proof Mode In this mode, the MACH network acts as a decentralised verifier network for validity proofs. The rollup operator such as a sequencer will commit to a newly set of transactions, the resulting state, together with the validity proof on MACH. The MACH network will then verify and reach consensus on the validity of the proof.

Despite the explicit usage of validity proofs, this mode can also work well with optimistic rollups. With optimistic rollups, any designated prover with the right incentive (outside of MACH), can generate a proof of validity and submit it to the MACH network which in turn verifies and reaches consensus on the validity of the proof. Note that for ZK rollups, the prover can generate and submit proofs more frequently on MACH than it does on Ethereum and it is important they do so for faster finality. Moreover, this does not need to happen at the expense of more proving work: Instead of waiting to create a single batch proof, the prover can create proofs in real time and send it to MACH and use recursion to aggregate them into a batch proof later that can go to Ethereum. As long as the incremental proofs are distributed right away to MACH, the transactions will experience fast finality. Previous Tier-3 Finality Next Interoperability via MACH Last modified2mo ago