Check out the rollup-boost README for instructions on how to get started and thevideo explainer of this post for an end-to-end tracing demo with rollup-boost.

I. Introduction

Rollup Boost is a new, highly modular sidecar solution designed for Optimism stack-based rollups, enabling them toaccess blocks from external block builders. Similar to MEV-Boost on Ethereum Layer 1 (L1), Rollup Boost allows Layer 2 (L2) rollups to interact with external block-building services without requiring modifications to the core OP stack software like op-node

or op-geth

. Rollup Boost is powering the upcoming Unichain

In this post, we'll delve into the technical architecture of Rollup Boost, exploring how it integrates with the Optimism stack, manages block-building processes, and introduces minimal latency overhead. We'll also provide a detailed walkthrough of an end-to-end tracing demo, showcasing the entire block-building lifecycle across a local OP stack devnet.

By the end, developers will have a solid understanding of how to leverage Rollup Boost in their rollup environments and observe outsourced block-building firsthand.

II. Understanding Rollup Boost's Architecture and Design Goals

A. High-Level Overview of Rollup Boost

At its core, Rollup Boost functions as a sidecar, providing Layer 2 rollups on the Optimism stack with the capability to interact with external block builders. It draws inspiration from MEV-Boost but is optimized for the unique requirements of the Optimism stack.

B. Key Design Goals

Simplicity

: Rollup Boost was designed with minimal complexity and maintaining a lightweight setup that avoids additional latency. By focusing on simplicity, Rollup Boost integrates smoothly into the Optimism stack without disrupting existing CL <> EL communication.

Modularity and Extensibility

: Recognizing that rollups have varying needs, Rollup Boost was built with extensibility in mind. It is designed to support custom block-building features, allowing operators to implement modifications for any variety of block selection or customization rules.

Liveness Protection

: To safeguard against liveness risks, Rollup Boost offers a local block production fallback if there is a failure in the external builder.

Compatibility

: Importantly, Rollup Boost allows operators to continue using standard op-node

or op-geth

software.

III. How Rollup Boost Works

A. Comparison to Ethereum L1 Block-Building Architecture

Understanding Rollup Boost requires a brief look at how block building is handled on Ethereum L1. In Ethereum's architecture, there are two main layers: the consensus layer (CL) and the execution layer (EL). These layers communicate via the Engine API, which serves two purposes:

1. Synchronization: The CL client updates the EL client with new blocks and chain status.
2. Block Building: The CL client triggers block building in the EL client.

In the block-building process, the CL client issues a fork choice update, specifying "payload attributes" that guide the EL client in generating a "payload ID." This ID is referenced to retrieve the block at the end of a slot.

B. Rollup Boost's Role in the Optimism Stack

The Optimism stack mirrors Ethereum's CL-EL architecture and reuses the Engine API. Rollup Boost leverages this setup by acting as a multiplexor, directing fork choice updates to both the local execution client and the builder's execution client simultaneously. This parallel operation streamlines block building across both clients. Rollup Boost also has the flexibility to function without a relay as it operates on the assumption of permissioned builders.

Boost Sync Feature

: The Boost Sync feature minimizes latency by routing synchronization calls directly to the builder's execution client. This step eliminates delays in the block-building process by allowing the block builder to start as soon as the proposer has issued a fork choice update. Testing has shown that this reduces latency significantly, from around 200-300 milliseconds to just 3-4 milliseconds in local environments i.e. no p2p networking overhead.

C. Security and Validation

In Rollup Boost, once the builder's block is submitted, it undergoes a validation step where Rollup Boost confirms its integrity by submitting a new payload call to the proposer's local execution client. This check ensures the builder's block is valid and prevents invalid blocks from affecting the rollup's liveness.

IV. Demo: Setting Up and Running Rollup Boost

A. Demo Prerequisites

To run the Rollup Boost demo, you'll need the following:

- Docker Compose

: For setting up and managing your development environment.

- Rollup Boost

: The Rollup Boost code and dependencies.

- OP Stack Components

: Including OP node and OP geth to simulate the Optimism environment.

B. Step-by-Step Demo Guide

Step 1

: Checkout the devnet docker compose setuphere to start the Optimism devnet.

Step 2

: Run make devnet-up

in the cloned repo. This step will initiate the network, enabling Rollup Boost to begin handling fork choice updates and block-building requests. To enable tracing and metrics, add --tracing

and --metrics

to the rollup-boost args in the docker-compose.yml

file under ops-bedrock/

Step 3

: Monitor Rollup Boost's operation with tracing logs with a distributed tracing platform likeJaeger. You should observe calls to fork choice updates, block retrieval, and new payload events as Rollup Boost coordinates with both the proposer's local client and the builder's execution client.

Step 4

: To shut down and reset the devnet back to a clean environment, run make devnet-down; make devnet-clean

C. Key Observations from Tracing

Using tracing, you'll gain insights into Rollup Boost's block-building lifecycle. Here's what you can expect to see:

- Fork Choice Updates (FCU)

: Each update triggers block-building, using payload attributes to specify the necessary block details.

- Latency Improvements

: Boost Sync reduces synchronization times with the builder significantly, allowing builders to start constructing blocks almost instantly following a fork choice update.

- Block Validation

: As each block is built, Rollup Boost validates the builder’s block through the local execution client, ensuring only valid blocks are considered.

V. Observability Features in Rollup Boost

Tracing and Metrics

Observability is a crucial feature in Rollup Boost’s latest release, enabling rollup operators to track the block-building lifecycle through logging, tracing and metrics. Tracing logs capture each step in block production, from payload creation to final block validation, offering operators a comprehensive view of Rollup Boost’s operations.