How to read the sequencer feed

Running an Arbitrum relay locally as a feed relay lets you subscribe to an uncompressed sequencer feed for real-time data as the sequencer accepts and orders transactions off-chain.

When connected to websocket port9642 of the local relay, you'll receive feed data that looks something like this:

{ "version": 1, "messages": [{ "sequenceNumber": 25757171, "message": { "message": { "header": { "kind": 3, "sender": "0xa4b000000000000000000073657175656e636572", "blockNumber": 16238523, "timestamp": 1671691403, "requestId": null, "baseFeeL1": null }, "l2Msg": "BAL40oKksUiElQL5AISg7rsAgxb6o5SZbYNoIF2DTixsqDpD2xII9GJLG4C4ZAhh6N0AAAAAAAAAAAAAAAAC7EQiq1R1VYgL3/oXgvD921hYRyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
}, "delayedMessagesRead": 354560 }, "signature": null }] } Breaking this down a bit: the top-level data structure is defined by the BroadcastMessage struct :

type BroadcastMessage struct { Version int json:"version" // Note: "Messages" is slightly ambiguous naming since there are different types of messages Messages []*BroadcastFeedMessage json:"messages,omitempty" ConfirmedSequenceNumberMessage* ConfirmedSequenceNumberMessage json:"confirmedSequenceNumberMessage,omitempty" } Themessages field is the BroadcastFeedMessage struct :

type BroadcastFeedMessage struct { SequenceNumber arbutil.MessageIndex json:"sequenceNumber" Message arbstate.MessageWithMetadata json:"message" Signature []byte json:"signature" } Eachmessage conforms to arbstate.MessageWithMetadata :

type MessageWithMetadata struct { Message *arbos.L1IncomingMessage json:"message" DelayedMessagesRead uint64 json:"delayedMessagesRead" } Finally, we get the transaction's information in themessage subfield as an L1IncomingMessage :

type L1IncomingMessage struct { Header *L1IncomingMessageHeader json:"header" L2msg []byte json:"l2Msg" // Only used for L1MessageType_BatchPostingReport BatchGasCost* uint64 json:"batchGasCost,omitempty" rlp:"optional" } In this file , we can find aParseL2Transactions function, which you can use this to decode the message.

Using the feed relay, you can also calculate the correspondingL2 block number

of a message:

On Arbitrum One, this can be done by adding the Arbitrum One genesis block number (22207817) to the sequence number of the feed message. It's important to note that in the case of Arbitrum Nova, the Nitro genesis number is 0, and therefore, there is no need to include it when adding to the feed message's sequence number. Edit this page Last updatedonMar 7, 2024 Previous How to run a feed relay Next How to run a Sequencer Coordinator Manager (SQM)