Sequencing, Followed by Deterministic Execution

This section explores the various methods users can employ to submit transactions for inclusion on the Arbitrum chain. We discuss the different pathways available—sending transactions to the Sequencer or bypassing it by submitting transactions through the Delayed Inbox contract on the Parent chain . By outlining these options, we aim to clarify how users can interact with the network, detail the processes involved in each method, and identify the modules responsible for handling these transactions. This overview will enhance your understanding of the initial steps in Arbitrum ecosystem's Transaction lifecycle and prepare you for a detailed exploration of transaction inclusion mechanisms in the subsequent sections.

The first subsection, Submitting Transactions to the Sequencer , presents four different methods users can utilize to send their transactions to the sequencer: via Public RPC, Third-Party RPC, Arbitrum Nodes, and the Sequencer Endpoint. Transactions sent through the first three pathways will route through our Load Balancer before reaching the sequencer. In contrast, the Sequencer Endpoint allows transactions to bypass the Load Balancer and be sent directly to the sequencer.

The second subsection, Bypassing the Sequencer , describes an alternative method where users can include their transactions on the Arbitrum chain without relying on the sequencer. By sending transactions directly to the delayed inbox contract on the parent chain (Layer 1), users gain additional flexibility, ensuring that their transactions can be processed even if the sequencer is unavailable or if they prefer not to use it.

This diagram illustrates the various pathways for submitting transactions to the Arbitrum chain. It highlights the options for sending transactions through the sequencer or bypassing it and using the delayed inbox contract on the parent chain.

Submitting transactions to the Sequencer

This section outlines the different methods for users to submit transactions to the sequencer on the Arbitrum chain. There are four primary ways to do this: Public RPC, Third-Party RPCs, Arbitrum Nodes, and the Sequencer Endpoint. We will explore these methods in detail, explaining when to choose one over the other and how to use each to effectively submit transactions to the Arbitrum sequencer.

1. Public RPC

Arbitrum provides public RPCs for its main chains: Arbitrum One , Arbitrum Nova , and Arbitrum Sepolia. Due to their rate-limited nature, these RPC endpoints are suitable for less resource-intensive operations. Public RPCs can be an accessible option for general use cases and light interactions with the network.

For more details on the specific RPC endpoints for each chain, please see this section of the documentation.

2. Third-Party RPC

Users also have the option to interact with Arbitrum's public chains using third-party node providers. These providers are often the same popular ones used for Ethereum, making them reliable choices for resource-intensive operations. We recommend using these third-party providers when performance and scalability are critical.

You can find a list of supported third-party providers here .

3. Arbitrum Nodes

Another approach for sending transactions to the sequencer is through self-hosted Arbitrum nodes. Running a node gives you direct control over your transactions, which go to the sequencer via the Sequencer Feed .

Please see the Arbitrum Node documentation to learn more about setting up and running a node.

4. Sequencer Endpoint

The Sequencer Endpoint is the most direct method for users looking to minimize delays in their transactions reaching the sequencer. Unlike standard RPC URLs, the Sequencer Endpoint supports only eth_sendRawTransaction and eth_sendRawTransactionConditional calls, bypassing the load balancer entirely. This endpoint makes it an optimal choice for users who require the quickest transaction processing time.

The diagram below shows different ways to submit transactions to the sequencer:

Bypassing the Sequencer

This section delves into an alternative method for submitting transactions to the Arbitrum chain, bypassing the sequencer. This page focuses on how users can send their transactions directly to the delayed inbox contract on the parent chain rather than through the sequencer. This method offers two distinct paths a transaction can take, with each route interacting with the network differently to achieve transaction inclusion. This approach provides users with greater flexibility and ensures that

transactions can still be processed if the sequencer is unavailable or if users choose not to depend on it. This section highlights these alternative submission mechanisms and underscores the robustness and decentralization features inherent in the Arbitrum network.

InDiagram 3 , we demonstrate how users can submit their transactions using the delayed inbox contract to bypass the sequencer. As illustrated in the diagram, there are two possible paths for transaction handling. When a transaction is submitted to the delayed inbox, the sequencer may automatically pick it up, include it as an ordered transaction, and send it to the sequencer feed. However, if the sequencer does not process the transaction within 24 hours, users have the reliable option to call theforceInclude function on the sequencer inbox contract. This action ensures that the sequencer to picks up the transaction and includes it in the ordered transaction list, providing users with a sense of security about their transactions.

To send a transaction to the delayed inbox instead of submitting it to the sequencer, users can construct their transaction and then call thesendL2Message function, passing the data of the serialized signed transaction as an argument. This function allows users to send a generic L2 message to the chain, suitable for any message that does not require L1 validation.

If the sequencer is not back online within 24 hours or decides to censor the transaction, users can invoke theforceInclusion function on the SequencerInbox contract. This action ensures their transaction is included on the chain, bypassing the sequencer's role.

Additionally, the Arbitrum SDK provides theInboxTools class, which simplifies the process of submitting transactions to the delayed inbox. Users can utilize thesendChildSignedTx method to send the transaction and theforceInclude method to ensure its inclusion. The SDK also offers helper methods likesignChildTx to assist with signing the transaction during the creation of the serialized signed transaction hex string, streamlining the entire process. Edit this page Last updatedonJan 27, 2025 Previous A gentle introduction Next The Sequencer and Censorship Resistance