External Node Documentation

On This Page * [Disclaimers *][What is the External Node? *][High-level Overview *][API Overview * * [eth_ Namespace * *][PubSub * *][net_ Namespace * *][web3_ Namespace * *][debug namespace * *][zks namespace * *][en namespace * *] *]

Info

For local testing, we recommend setting up an in-memory node and forking mainnet.

# #

External Node Documentation

This documentation explains the basics of the zkSync Era External Node.

#

Disclaimers

- The external node is in the alpha phase, and should be used with caution.
- While in alpha, the EN is dependent on a DB snapshot in order to run that is not yet publicly available.
- The EN is a read-only replica of the main node. We are currently working on decentralizing our infrastructure by creating a consensus node. The EN is not going to be the consensus node.

#

What is the External Node?

The external node (herein EN) is a read-replica of the main (centralized) node that can be run by external parties. It functions by fetching data from the zkSync API and re-applying transactions locally, starting from the genesis block. The EN shares most of its codebase with the main node. Consequently, when it re-applies transactions, it does so exactly as the main node did in the past.

In Ethereum terms, the current state of the EN represents an archive node, providing access to the entire history of the blockchain.

#

High-level Overview

At a high level, the EN can be seen as an application that has the following modules:

- API server that provides the publicly available Web3 interface.
- Synchronization layer that interacts with the main node and retrieves transactions and blocks to re-execute.
- Sequencer component that actually executes and persists transactions received from the synchronization layer.
- Several checker modules that ensure the consistency of the EN state.

With the EN, you are able to:

- Locally recreate and verify the zkSync Era mainnet/testnet state.
- Interact with the recreated state in a trustless way (in a sense that the validity is locally verified, and you should not rely on a third-party API zkSync Era provides).
- Use the Web3 API without having to query the main node.
- Send L2 transactions (that will be proxied to the main node).

With the EN, youcan not :

- Create L2 blocks or L1 batches on your own.
- Generate proofs.
- Submit data to L1.

A more detailed overview of the EN's components is provided in the components section.

#

API Overview

API exposed by the EN strives to be Web3-compliant. If some method is exposed but behaves differently compared to Ethereum, it should be considered a bug. Please[reportopen in new window]such cases.

[#]

eth_ Namespace

Data getters in this namespace operate in the L2 space: require/return L2 block numbers, check balances in L2, etc.

Available methods:

Method Notes eth_blockNumber eth_chainId eth_call eth_estimateGas eth_gasPrice eth_newFilter Maximum amount of installed filters is configurable eth_newBlockFilter Same as above eth_newPendingTransactionsFilter Same as above eth_uninstallFilter eth_getLogs Maximum amount of returned entities can be configured eth_getFilterLogs Same as above eth_getFilterChanges Same as above eth_getBalance eth_getBlockByNumber eth_getBlockByHash eth_getBlockTransactionCountByNumber eth_getBlockTransactionCountByHash eth_getCode eth_getStorageAt eth_getTransactionCount eth_getTransactionByHash eth_getTransactionByBlockHashAndIndex eth_getTransactionByBlockNumberAndIndex eth_getTransactionReceipt eth_protocolVersion eth_sendRawTransaction eth_syncing EN is considered synced if it's less than 11 blocks behind the main node. eth_coinbase Always returns a zero address eth_accounts Always returns an empty list eth_getCompilers Always returns an empty list eth_hashrate Always returns zero eth_getUncleCountByBlockHash Always returns zero eth_getUncleCountByBlockNumber Always returns zero eth_mining Always returns false

[#]

PubSub

Only available on the WebSocket servers.

Available methods:

Method Notes eth_subscribe Maximum amount of subscriptions is configurable eth_subscription

[#]

net_ Namespace

Available methods:

Method Notes net_version net_peer_count Always returns 0 net_listening Always returns false

[#]

web3_ Namespace

Available methods:

Method Notes web3_clientVersion

[#]

debug namespace

Thedebug namespace gives access to several non-standard RPC methods, which will allow developers to inspect and debug calls and transactions.

This namespace is disabled by default and can be configured via settingEN_API_NAMESPACES as described in the example config.

Available methods:

Method Notes debug_traceBlockByNumber debug_traceBlockByHash debug_traceCall debug_traceTransaction

[#]

zks namespace

This namespace contains rollup-specific extensions to the Web3 API. Note thatonly methods specified in the documentation are considered public. There may be other methods exposed in this namespace, but undocumented methods come without any kind of stability guarantees and can be changed or removed without notice.

Always refer to the documentation linked above to see the list of stabilized methods in this namespace.

zks_docs

#

en namespace

This namespace contains methods that external nodes call on the main node while syncing. If this namespace is enabled other ENs can sync from this node.

Next Component Breakdown