Setting up a Celestia full storage Node

This tutorial will guide you through setting up a Celestia full storage node, which is a celestia-node that doesn't connect to celestia-app (hence not a consensus node), but stores all the data.

Overview of full storage nodes

Full storage nodes are Celestia nodes that store all the data. Full storage nodes send block shares, headers, and fraud proofs to light nodes. The light nodes gossip headers, fraud proofs, and sometimes block shares, between one another.

Hardware requirements

The following hardware minimum requirements are recommended for running the full storage node:

- Memory:16 GB RAM (minimum)
- CPU:6 cores
- Disk:2 TB NVME Storage
- Bandwidth:1 Gbps for Download/1 Gbps for Upload

Setting up your full storage node

The following tutorial is done on an Ubuntu Linux 20.04 (LTS) x64 instance machine.

Setup the dependencies

You can followthe tutorial for setting up your dependencies

Install celestia-node

You can followthe tutorial for installingcelestia-node

Run the full storage node

Initialize the full storage node

Run the following command:

Mainnet Beta

Mocha

Arabica sh celestia

full

init celestia

full

init sh celestia

full

init

--p2p.network

mocha celestia

full

init

--p2p.network

mocha sh celestia

full

init

--p2p.network

arabica celestia

full

init

--p2p.network

arabica

Start the full storage node

Start the full storage node with a connection to a validator node's gRPC endpoint (which is usually exposed on port 9090):

In order for access to the ability to get/submit state-related information, such as the ability to submitPayForBlob transactions, or query for the node's account balance, a gRPC endpoint of a validator (core) node must be passed as directed below.

Refer tothe ports section of the celestia-node troubleshooting page for information on which ports are required to be open on your machine.

sh celestia

full

start

--core.ip

< UR I

celestia

full

start

--core.ip

< UR I

Using an RPC of your own, or one fromMainnet Beta ,Mocha testnet orArabica devnet , start your node.

Connecting to a core endpoint with--core.ip string provides the light node with access to state queries (reading balances, submitting transactions, and other state-related queries).

You can create your key for your node by followingthecel-key instructions

Once you start the full storage node, a wallet key will be generated for you. You will need to fund that address with testnet tokens to pay forPayForBlob transactions. You can find the address by running the following command:

sh ./cel-key

list

--node.type

full

--keyring-backend

test

--p2p.network

< networ k

./cel-key

list

--node.type

full

--keyring-backend

test

--p2p.network

< networ k

TIP

You do not need to declare a network for Mainnet Beta. Refer to the chain ID section on the troubleshooting page for more information You can get testnet tokens from:

- Mocha
- Arabica

NOTE

If you are running a full-storage node for your sovereign rollup, it is highly recommended to request Arabica devnet tokens as Arabica has the latest changes that can be used to test for developing your sovereign rollup. You can still use Mocha testnet as well, it is just mostly used for validator operations.

Optional: run the full storage node with a custom key

In order to run a full storage node using a custom key:

1. The custom key must exist inside the celestia full storage node directory at the correct path (default:~/.celestia-full/keys/keyring-test
2.)
3. The name of the custom key must be passed uponstart
4. , like so:

Mainnet Beta

Mocha

Arabica sh celestia

full

start

--core.ip

< UR I

\ --keyring.keyname

< name-of-custom-ke y

\ celestia

full

start

--core.ip

< UR I

\ --keyring.keyname

< name-of-custom-ke y

\ sh celestia

full

start

--core.ip

< UR I

\ --keyring.keyname

< name-of-custom-ke y

\ --p2p.network

mocha celestia

full

start

--core.ip

< UR I

\ --keyring.keyname

< name-of-custom-ke y

\ --p2p.network

mocha sh celestia

full

start

--core.ip

< UR I

\ --keyring.keyname

< name-of-custom-ke y

\ --p2p.network

arabica celestia

full

start

--core.ip

< UR I

\ --keyring.keyname

< name-of-custom-ke y

\ --p2p.network

arabica

Optional: Migrate node id to another server

To migrate a full storage node ID:

1. You need to back up two files located in the celestia-full node directory at the correct path (default:~/.celestia-full/keys
2.).
3. Upload the files to the new server and start the node.

Optional: start the full storage node with SystemD

If you would like to run the full storage node as a background process, follow the SystemD tutorial .

With that, you are now running a Celestia full storage node.

Stop the full storage node

In order to gracefully stop the full storage node, useControl + C in the terminal window where the node is running. Be sure to only do this once as the shutdown will not be instantaneous. []

[[Edit this page on GitHub] Last updated: Previous page Light node Next page Bridge node []