Running an OP Mainnet Node from Source

This tutorial explains how to run an OP Mainnet node from source code for full nodes and archive nodes. Running an OP Mainnet node from source code is a flexible alternative to using pre-built Docker images.

Building the Source Code

You'll need to buildop-node andop-geth from their respective source repositories before you can run a node. Make sure to follow the instructions onBuilding a Node from Source before continuing.

Hardware Requirements

Hardware requirements for OP Mainnet nodes can vary depending on the type of node you plan to run. Archive nodes generally require significantly more resources than full nodes. Below are suggested minimum hardware requirements for each type of node.

- 16GB RAM
- 1TB SSD (full node) or 4TB SSD (archive node)
- Reasonably modern CPU

Full Nodes

Assess Blob Archiver

Assess if you need to configure a blob archiver service by reading theConfigure a Blob Archiver documentation .

Create a JWT Secret

op-geth andop-node communicate over the engine API authrpc. This communication is secured using a shared secret. You will need to generate a shared secret and provide it to bothop-geth andop-node when you start them. In this case, the secret takes the form of a 32 byte hex string.

Run the following command to generate a random 32 byte hex string:

openssl

rand

-hex

32

jwt.txt

Startop-geth

It's generally easier to startop-geth before startingop-node . You can still startop-geth without yet runningop-node , but theop-geth instance will simply not receive any blocks untilop-node is started.

Navigate to your op-geth directory

Find the directory where you built theop-geth binary.

Copy in the JWT secret

Copy the JWT secret you generated in a previous step into theop-geth directory.

cp

/path/to/jwt.txt

.

Set environment variables

Set the following environment variables:

export DATADIR_PATH = ...

Path to the desired data directory for op-geth

Start op-geth

Use the following command to startop-geth in a default configuration. The JSON-RPC API will become available on port 8545. Refer to theop-geth configuration documentation for more detailed information about available options.

./build/bin/geth \ --http \ --http.port=8545 \ --http.addr=localhost \ --authrpc.addr=localhost \ --authrpc.jwtsecret=./jwt.txt \ --verbosity=3 \ --rollup.sequencerhttp=https://mainnet-sequencer.optimism.io/ \ --op-network=op-mainnet \ --datadir=DATADIR_PATH

Startop-node

Once you've startedop-geth , you can startop-node .op-node will connect toop-geth and begin synchronizing the OP Mainnet state.op-node will begin sending block payloads toop-geth when it derives enough blocks from Ethereum.

Navigate to your op-node directory

Find the directory where you built theop-node binary.

Copy in the JWT secret

Bothop-geth andop-node need to use the same JWT secret. Copy the JWT secret you generated in a previous step into theop-node directory.

cp

/path/to/jwt.txt

.

Set environment variables

Set the following environment variables:

export L1_RPC_URL = ...

URL for the L1 node to sync from

export L1_RPC_KIND = ...

RPC type (alchemy, quicknode, infura, parity, nethermind, debug_geth, erigon, basic, any)

export L1_BEACON_URL = ...

URL address for the L1 Beacon-node HTTP endpoint to use.

Start op-node

Use the following command to startop-node in a default configuration. Refer to theop-nodeconfiguration documentation for more detailed information about available options.

⚠ Theop-node RPC should not be exposed publicly. If left exposed, it could accidentally expose admin controls to the public internet. ./bin/op-node \ --l1=L1_RPC_URL \ --l1.rpckind=L1_RPC_KIND \ --l1.beacon=L1_BEACON_URL \ --l2=ws://localhost:8551 \ --l2.jwt-secret=./jwt.txt \ --network=op-mainnet \ --syncmode=execution-layer
Some L1 nodes, like Erigon, do not support theeth_getProof RPC method that theop-node uses to load L1 data for certain processing steps. If you are using an L1 node that does not supporteth_getProof , you will need to include the--l1.trustrpc flag when startingop-node . Note that this flag will causeop-node to trust the L1 node to provide correct data as it will no

longer be able to independently verify the data it receives.

Synchronization Verification

Once you've startedop-geth andop-node you should see the two begin to communicate with each other and synchronize the OP Mainnet chain.

Snap Sync (Default)

Initial synchronization can take several hours to complete. You will see theseop-node logs at the start of snap sync:

INFO [03-06|10:56:55.602] Starting EL sync INFO [03-06|10:56:55.615] Sync progress reason="unsafe payload from sequencer while in EL sync" l2_finalized=000000..000000:0 l2_safe=000000..000000:0 l2_pending_safe=000000..000000:0 l2_unsafe=4284ab..7e7e84:117076319 l2_time=1,709,751,415 l1_derived=000000..000000:0 INFO [03-06|10:56:57.567] Optimistically inserting unsafe L2 execution payload to drive EL sync id=4ac160..df4d12:117076320 Starting EL sync is shown once and thesync progress / inserting logs should be repeated until done.

op-node will log the following when done:

lvl=info msg="Finished EL sync" sync_duration=23h25m0.370558429s finalized_block=0x4f69e83ff1407f2e2882f2526ee8a154ac326590799889cede3af04a7742f18d:116817417 There are two stages onop-geth for snap sync:

Downloading the headers

op-geth log something like this as it is downloading the headers:

lvl=info msg="Syncing beacon headers" downloaded=116775778 left=1162878 eta=53.182s

Sync progress

For the second stage,op-geth will log the following:

lvl=info msg="Syncing: state download in progress" synced=99.75% state="191.33 GiB" accounts=124,983 slots=806,829, eta=-2m7.602s msg="Syncing: chain download in progress" synced=100.00% chain="176.01 GiB" headers=116,817, bodies=116,817, receipts=116,817, eta=77.430ms All the while,op-geth will also log the forkchoice update:

Forkchoice requested sync to new head number=117,076,468 hash=e3884c..bf4e2b

Full Sync

Initial full synchronization can take several days or weeks to complete.

During this time, you will initially observeop-node deriving blocks from Ethereum without sending these blocks toop-geth . This means thatop-node is requesting blocks from Ethereum one-by-one and determining the corresponding OP Mainnet blocks that were published to Ethereum. You should see logs like the following fromop-node :

INFO [06-26|13:31:20.389] Advancing bq origin origin=17171d..1bc69b:8300332 originBehind=false Once theop-node has derived enough blocks from Ethereum, it will begin sending these blocks toop-geth . You should see logs like the following fromop-node :

INFO [06-26|14:00:59.460] Sync progress reason="processed safe block derived from L1" l2_finalized=ef93e6..e0f367:4067805 l2_safe=7fe3f6..900127:4068014 l2_unsafe=7fe3f6..900127:4068014 l2_time=1,673,564,096 l1_derived=6079cd..be4231:8301091 INFO [06-26|14:00:59.460] Found next batch epoch=8e8a03..11a6de:8301087 batch_epoch=8301087 batch_timestamp=1,673,564,098 INFO [06-26|14:00:59.461] generated attributes in payload queue txs=1 timestamp=1,673,564,098 INFO [06-26|14:00:59.463] inserted block hash=e80dc4..72a759 number=4,068,015 state_root=660ced..043025 timestamp=1,673,564,098 parent=7fe3f6..900127 prev_randao=78e43d..36f07a fee_recipient=0x4200000000000000000000000000000000000011 txs=1 update_safe=true You should then also begin to see logs like the following fromop-geth :

INFO [06-26|14:02:12.974] Imported new potential chain segment number=4,068,194 hash=a334a0..609a83 blocks=1 txs=1 mgas=0.000 elapsed=1.482ms mgasps=0.000 age=5mo2w20h dirty=2.31MiB INFO [06-26|14:02:12.976] Chain head was updated number=4,068,194 hash=a334a0..609a83 root=e80f5e..dd06f9 elapsed="188.373µs" age=5mo2w20h INFO [06-26|14:02:12.982] Starting work on payload id=0x5542117d680dbd4e

Archive Nodes

You only need an archive node if you need the historical state. Most node operators should default to full nodes.

Get the Migrated Data Directory

OP Mainnet underwent a large database migration as part of theBedrock Upgrade(opens in a new tab) in 2023. You will need access to the migrated OP Mainnet database to run an archive node. You canmigrate your own data directory(opens in a new tab) or simply download database that has already been migrated. In this section, you'll learn how to download and verify the pre-migrated database.

Download the Migrated Data Directory

Click the link below to find the latest publicly available database snapshots for OP Mainnet. Snapshots are available for multiple dates and snapshots get larger as they get closer the current date. Snapshots are large files and may take some time to download.OP Mainnet Snapshots

Verify the Download

You should always verify the integrity of your downloads to ensure that they have not been corrupted. A corrupted database can include invalid data or may cause your node to fail. Verify the integrity of the download by checking the SHA256 checksum of the downloaded file.

sha256sum

< filenam e

> For instance, if you've downloaded the very first database snapshot, you can verify the download by running the following command:

sha256sum

mainnet-bedrock.tar.zst You should see then following output:

ec4baf47e309a14ffbd586dc85376833de640c0f2a8d7355cb8a9e64c38bfcd1

mainnet-bedrock.tar.zst Your exact output will depend on the snapshot you've downloaded. Check theOP Mainnet Snapshots page for the correct checksum for the snapshot you've downloaded.

Extract the Data Directory

Once you've downloaded the database snapshot, you'll need to extract it to a directory on your machine. This will take some time to complete.

tar

xvf

< filenam e

> For instance, if you've downloaded the very first database snapshot, you can extract it by running the following command:

tar

xvf

mainnet-bedrock.tar.zst

Configure op-geth for archive mode

Set--syncmode=full and--gcmode=archive onop-geth .

Get the Legacy Geth Directory (Optional)

Blocks and transactions included in OP Mainnet before the Bedrock Upgrade cannot be executed by modern OP Mainnet nodes. OP Mainnet nodes willserve these blocks and transactions but cannot run certain queries against them (e.g.eth_call). If you need to run stateful queries likeeth_call against these older blocks and transactions, you will need to run a Legacy Geth node alongside your OP Mainnet node.

Running a Legacy Geth node is entirely optional and typically only useful for operators who want to run complete archive nodes of the OP Mainnet state. If you want to run a full node then you can safely skip this section.

Download the Legacy Geth Data Directory

Click the link below to download the latest publicly available database snapshot for Legacy Geth. This is a very large file (2.9TB), so expect the download to take some time to complete.

Legacy Geth Data Directory (2.9TB)

Verify the Download

You should always verify the integrity of your downloads to ensure that they have not been corrupted. A corrupted database can include invalid data or may cause your node to fail. Verify the integrity of the download by checking the SHA256 checksum of the downloaded file.

sha256sum

mainnet-legacy-archival.tar.zst You should see the following output:

4adedb61125b81b55f9bdccc2e85092050c65ef2253c86e2b79569732b772829

mainnet-legacy-archival.tar.zst If you see a different output, then the download is corrupted and you should try downloading the file again.

Extract the Data Directory

Once you've downloaded the database snapshot, you'll need to extract it to a directory on your machine. This will take some time to complete.

tar

xvf

mainnet-legacy-archival.tar.zst

Start Legacy Geth (Optional)

If you've chosen to run a Legacy Geth node alongside your OP Mainnet node, you'll need to start it before you start your OP Mainnet node.

Navigate to your Legacy Geth directory

Find the directory where you built thel2geth binary.

Start l2geth

Run the following command to startl2geth :

USING_OVM

true \ ETH1_SYNC_SERVICE_ENABLE = false \ RPC_API = eth,rollup,net,web3,debug \ RPC_ENABLE = true \ RPC_PORT = 8546 \ ./build/bin/geth

--datadir

/path/to/l2geth-datadir

Next Steps

- If you've already got your node up and running, check out theNode Metrics and Monitoring Guide
- to learn how to keep tabs on your node and make sure it keeps running smoothly.
- If you run into any problems, please visit theNode Troubleshooting Guide
- for help.

Building a Node from Source Running OP Sepolia from Source