Running the External Node

#

Running the External Node

This section assumes that you have prepared a configuration file as described on the previous page.

#

Preferred hardware configuration

This configuration is approximate, expect updates to these specs.

- 32-core CPU
- 64GB RAM
- SSD storage:* Testnet - ~800 GB (at the time of writing) and will grow over time, so should be constantly monitored
-
    - Mainnet - ~400 GB (at the time of writing) and will grow over time, so should be constantly monitored
-
    - NVMe recommended
- 100 Mbps network connection.

#

A note about PostgreSQL storage

By far, the heaviest table to maintain is thecall_traces table. This table is only required for thedebug namespace. If you want to clear some space and aren't using thedebug namespace, you can

- clear it with a simple queryDELETE FROM call_traces;
- leave thedebug
- namespace disabled via theEN_API_NAMESPACES
- env var as described in the example config.

#

Infrastructure

You need to set up a PostgreSQL server with SSD storage:

- Testnet - ~1TB (at the time of writing) and will grow over time, so should be constantly monitored
- Mainnet - ~2TB (at the time of writing) and will grow over time, so should be constantly monitored

Setting up Postgres is out of the scope of these docs, but the popular choice is to run it in Docker. There are many of guides on that,here's one exampleopen in new window.

Note however that if you run Postgres as a stand-alone Docker image (e.g. not in Docker-compose with a network shared between EN and Postgres), EN won't be able to access Postgres vialocalhost or127.0.0.1 URLs. To make it work, you'll have to either run it with a--network host (on Linux) or usehost.docker.internal instead oflocalhost in the EN configuration (official docs).

Besides running Postgres, you are expected to have a DB dump from a corresponding env. You can restore it usingpg_restore -O -C --dbname= .

host_docker_internalopen in new window

#

Running

Assuming you have the EN Docker image, an env file with the prepared configuration, and you have restored your DB with the pg dump, that is all you need.

Sample running command:

docker run --env-file< path_to_env_file> --mount type = bind,source= < local_rocksdb_data_path> ,target= < configured_rocksdb_data_path> < image> Helm charts and other infrastructure configuration options, if required, would be available later.

#

First start

When you start the node for the first time, the state in PostgreSQL corresponds to the dump you have used, but the state in RocksDB (mainly the Merkle tree) is absent. Before the node can make any progress, it has to rebuild the state in RocksDB and verify consistency. The exact time required for that depends on the hardware configuration, but it is reasonable to expect the state rebuild on the mainnet to take more than 20 hours.

#

Redeploying the EN with a new PG dump

If you've been running the EN for some time and are going to redeploy it using a new PG dump, you should

- Stop the EN
- Remove SK cache (corresponding toEN_STATE_CACHE_PATH
-)
- Remove your current DB
- Restore with the new dump
- Start the EN

Monitoring the node behavior and analyzing the state it's in is covered in the observability section.

Prev Configuration Next API Overview