Database

Nethermind uses theRocksDB database to store the state. By default, the database is located in the same directory where the Nethermind executable is. You can change the database location using the-d, --baseDbPath command line option.

Database directory structure

Directory Description blockInfos Information about blocks at each level of the block tree (canonical chain and branches) blocks Block bodies (block transactions and uncles) bloom Bloom indices for fast log searches canonicalHashTrie LES protocol related data code Contract bytecodes discoveryNodes Peers discovered via discovery protocol - used for quick peering after restarts (you can copy this DB between nodes to speed up peering) headers Block headers only pendingTx The second level cache of pending transactions/mempool (the first level is in memory). Wiped out on each restart. peers Additional sync peers information (like peer reputation) - you can copy this DB between nodes to speed up peering on fresh sync receipts Transaction receipts state Blockchain state including accounts and contract storage (Patricia trie nodes) You can usersync between your nodes to clone the database (One of our users copied the entire 4.5TB archive state this way while the node was running and only stopped the node for the very last stage ofrsync). You can also copy the database between Linux, Windows, and macOS.

Database size

The table below presents the Nethermind database size after fresh sync with different configurations and networks (as of May 2023).

Network/configuration Database size after fresh sync Mainnet (default) 898 GB Mainnet with ancient barriers 738 GB Mainnet archive 14 TB Mainnet without old bodies and receipts 178 GB Gnosis 112 GB Goerli 240 GB

Database size in details

Below is a comprehensive list of the supported chains, along with a detailed breakdown of their respective database directories. For reference, the database sizes listed are based on the data from July 2023 and have been determined using the standard configurations provided.

- Mainnet
- Goerli
- Sepolia
- Gnosis
- Chiado
- Energy Web
- Volta

Database Mainnet Mainnet with ancient barriers State 181 GB 181 GB Receipts 203 GB 155 GB Blocks 497 GB 385 GB Bloom 5.9 GB 5.9 GB Headers 8.1 GB 8.1 GB Code 3.9 GB 3.9 GB Other ... ... Total 898 GB 738 GB Directory Size state 51 GB receipts 35 GB blocks 185 GB code 7.4 GB peers 8 KB witness 4.1 MB blockInfos 417 MB bloom 2.5 GB headers 3.5 GB canonicalHashTrie 4.1 MB metadata 4.4 MB discoveryNodes 1.5 MB TOTAL 283 GB Directory Size state 8.5 GB receipts 8.2 GB blocks 36 GB code 1.5 GB peers 8 KB witness 4.1 MB blockInfos 177 MB bloom 1.3 GB headers 1.2 GB canonicalHashTrie 4.1 MB metadata 4.1 MB discoveryNodes 1.7 MB TOTAL 56 GB Directory Size state 40 GB receipts 21 GB blocks 43 GB code 350 MB peers 8 KB witness 4.1 MB blockInfos 1.5 GB bloom 7.5 GB headers 9.4 GB canonicalHashTrie 4.1 MB metadata 4.2 MB discoveryNodes 1.8 MB TOTAL 121 GB Directory Size state 1.6 GB receipts 646 MB blocks 2.3 GB code 22 MB peers 8 KB witness 4.1 MB blockInfos 224 MB bloom 1.4 GB headers 1.1 GB canonicalHashTrie 4.1 MB metadata 4.1 MB discoveryNodes 1.7 MB TOTAL 7.1 GB Directory Size state 31.13 GB receipts 5.13 GB blocks 15.55 GB code 7.20 MB peers 311 B witness 103.11 KB blockInfos 1.22 GB bloom 7.78 GB headers 6.70 GB canonicalHashTrie 103.35 KB metadata 103.31 KB discoveryNodes 1.68 MB TOTAL 67.52 GB Directory Size state 35.71 GB receipts 7.67 GB blocks 25.11 GB code 75.14 MB peers 616 B witness 119.95 KB blockInfos 1.15 GB bloom 7.40 GB headers 6.94 GB canonicalHashTrie 120.00 KB metadata 119.97 KB discoveryNodes 1.55 MB TOTAL 84.06 GB

Reducing database size

The Nethermind database can experience substantial growth over time, starting from an initial size of approximately 650 GB. As a result, many node setups are configured to run on 1 TB disks. However, even with settings designed to slow the growth rate, these disks may eventually run out of free space.

The current options to reduce the database size are as follows:

- Resyncing database from scratch
- Pruning

The table below presents a short comparison of these methods including possible fine-tuning of each method. Data was

fetched from a node running on a machine with the below specifications:

- Node.js
- : v1.18.0
- Consensus client
- : Lighthouse
- CPU
- : AMD EPYC 7713 (16 cores allocated for the VM)
- RAM
- : 64 GB
- Disk size
- : 1.2 TB
- Disk IOPS
- : 70,000 to 80,000

Metric Resync Pruning Pruning and memory budget (4 GB) Execution time ~4h ~24h ~12h Minimum free disk space N/A. You can execute resync even if there is 0 free space (avoid such a case). 250 GB 250 GB Attestation rate drop 100%. No attestation rewards during that time or highly reduced. 5–10% during that time N/A Average block processing time of new blocks during the process N/A. New blocks are processed after state but are significantly slower until old bodies/receipts are downloaded. Afterward, average about 0.35s. 0.7s 1.0s Is the node online during the process? No, unless the state is synced. Yes. The node follows the chain, and all modules are still enabled. Yes. The node follows chain and all modules are still enabled. The command used for testing disk IOPS was as follows:

fio --randrepeat = 1

--ioengine

libaio --direct = 1

--gtod_reduce

1

--name

test --filename = test --bs = 4k --iodepth = 64

--size

4G --readwrite = randrwEdit this page Last updatedonFeb 17, 2024 Previous Logs Next Pruning