Performance tuning

By default, Nethermind is configured for general use cases that fit well for most users. However, to improve various aspects of Nethermind performance, there are options for different subsystems that can be configured for your specific needs.

Peer discovery

To connect to the Ethereum network, Nethermind needs to maintain connections to other clients. The number of connections can be configured with--Network.MaxActivePeers . The default value depends on the network. Increasing this number may reduce syncing time, while reducing this number may help with attestation performance. Also, you can increase the rate at which a new connection is established with--Network.MaxOutgoingConnectsPerSec . The default value is 20 while 50 would be a reasonable higher value. This tends to reduce the snap sync time; however, some ISPs may throttle your Internet connection if you set this value too high. Also, some WiFi routers may hang if the value is set too high.

Port forwarding

While port forwarding is not strictly required, it helps significantly with finding peers and is essential for the network's overall health. The exact steps for port forwarding highly depend on your environment, router, and ISP. For most home configurations, automatic port forwarding can be turned on with--Network.EnableUPnP true . Some ISPs are more restrictive and do not support port forwarding and/or utilize provider-level NAT. In such cases, your best option is to use a VPN that supports port forwarding. Keep in mind that consensus clients need a separate port forwarding.

Sync time

On the Ethereum mainnet, most of the syncing time is split into three phases: snap sync, old bodies, and old receipts. Strictly speaking, there are also fast sync, full sync, and state sync phases. However, they usually complete in less than a minute, with state sync usually taking up to 3 minutes.

At the moment, the best test case sync time is 1 hour 50 minutes for all phases with the following configuration:

- CPU: AMD Ryzen 9 7950X
- Memory: 128GB RAM
- Storage: Intel Optane SSD 905P Series 900GB
- Network: 1 Gbps Internet with TorGuard VPN with WireGuard protocol. Both execution and consensus clients port forwarding are set up manually.
- Command line options:
- --Network.EnableUPnP true
- --Network.MaxOutgoingConnectsPerSec 50
- --Network.ProcessingThreadCount 32
- --Sync.TuneDbMode HeavyWrite

Snap sync

Snap sync is the process of downloading the Ethereum state tree. After it is complete, and after the state sync phase, Nethermind can process and follow the chain. The fastest tested snap sync and state sync time is 25 minutes. This phase is the most I/O-intensive sync phase, and therefore, assuming a fast internet, the sync time highly depends on your SSD's write speed. Remember that most SSDs only advertise peak write speed, usually above 5GB/s. However, they tend to slow down significantly to around 0.5GB/s (or even less for a QLC SSD) after a few seconds. Therefore, look for SSDs with high sustained write speed.

Also, ensuring your SSD is sufficiently cooled to prevent thermal throttling is essential. This is often overlooked as most workloads rarely stress SSD as much; however, to reduce sync time, Nethermind will utilize your SSD to its limit. If, for whatever reason, you need to minimize the I/O load, you can specify a rate limit with--Db.MaxBytesPerSec 1000000000 .

Nethermind temporarily changes the database configuration during sync to optimize it for writing, notably the option--Sync.TuneDbMode HeavyWrite is turned on by default. On some systems with slow SSDs, the use of the option--Sync.TuneDbMode AggressiveHeavyWrite may boost. Also, the option--Sync.TuneDbMode DisableCompaction can be used to disable compaction altogether. This is likely faster for systems using entry-level NVMe SSDs and is also useful to extend the lifespan of your SSD as it provides the lowest total writes possible. However, it uses about 3GB of extra memory during snap sync. The state sync phase may appear to hang for about 10 minutes as the whole database compacts for the first time after snap sync.

If you are running on a VPS with artificially capped IOPS, or you are using SATA SSD (which is highly not recommended), increasing the state DB block size with--Db.StateDbBlockSize 16384 may help to reduce snap sync time. However, this negatively affects block processing time. An alternative is to turn on compaction readahead with--Db.CompactionReadAhead 128000 ; however, this may take up a few extra GB of memory depending on the readahead

value.

Old bodies and receipts

Old bodies and old receipts are the process of downloading block bodies and receipts. This is required for some RPC methods, such aseth_getLogs , and for consensus clients to work correctly. If you don't need them, skip this phase with

--Sync.DownloadBodiesInFastSync false --Sync.DownloadReceiptsInFastSync false --Sync.NonValidatorNode true Old bodies and receipts are mainly limited by your Internet connection. With a 1Gbps connection, they consume around 250MB/s and 500MB/s of writes, respectively, which is generally reasonable for most PCIE SSDs. On older systems or VPS with low single thread performance and high Internet speed, the block body deserialization may be a bottleneck, in which case, you can increase the number of network processing threads with--Network.ProcessingThreadCount 32 . However, this may impact block processing time.

Block processing time and attestation

Block processing time is limited mainly by SSD performance. Strictly speaking, it's not the IOPS that matters, but the response time. Nevertheless, the IOPS is a good approximation as most SSDs don't advertise the response time. To help further reduce reads from SSD, Nethermind has multiple levels of caching, which is tuned by the memory hint option--Init.MemoryHint 2000000000 . If you are running a system with more than 16GB of memory, it is highly recommended to increase this value. In-memory pruning (turned on by default) also improves block processing time.

It is also possible to disable compression of the state DB with--Db.StateDbDisableCompression true that improves block processing time by 3% to 5% but increases disk space usage correspondingly. Block processing is susceptible to the number of peers connected. Therefore, after the node is synced, it makes sense to reduce the number of peers with--Network.MaxActivePeers 20 . Edit this page Last updatedonMar 26, 2024 Previous Pruning Next Private networks