Mantlemint

Note: Mantlemint is currently in beta. This means some of these instructions may not work as expected, or could be subject to change

What is Mantlemint?

Mantlemint is a fast core optimized for serving massive user queries. A mantlemint node will perform 3-4x more queries than a standard Secret Node.

Native query performance on RPC is slow and is not suitable for massive query handling, due to the inefficiencies introduced by IAVL tree. Mantlemint is running onfauxMerkleTree mode, basically removing the IAVL inefficiencies while using the same core to compute the same module outputs.

If you are looking to serve any kind of public node accepting varying degrees of end-user queries, it is recommended that you run a mantlemint instance alongside of your RPC. While mantlemint is indeed faster at resolving queries, due to the absence of IAVL tree and native tendermint, it cannot join p2p network by itself. Rather, you would have to relay finalized blocks to mantlemint, using RPC's websocket.

Mantlemint has been adapted for Secret Network from Terra.

Features

- Superior LCD performance
-
    - With the exception of Tendermint RPC/Transactions.
- *
- Super reliable and effective LCD response cache to prevent unnecessary computation for query resolving
- Fully archival; historical states are available with?height
- query parameter.
- Useful default indexes
-

Prerequisites

1. Fully synced RPC Node with Websockets available
2.

To start a Mantlemint node, you'll need **** access to at least 1 running RPC node. Since Mantlemint cannot join p2p network by itself, it depends on RPC to receive recently proposed blocks. This RPC node should also have Websockets enabled. Websockets are how Mantlemint receives new blocks after it catches up to the current block.

Minimum Requirements

1. 1TB of storage (recommended SATA or NVMe SSD)
2. 16 GB of RAM (recommended 32 GB)
3. 2 available CPU cores (recommended 4 cores)
4.

Mantlemint can only be ran as a full archive node. For this reason it requires a large amount of storage (as of August 2022 this is currently 450GB).

Setup

1a. Build from source

a. Clone the repositoryhttps://github.com/scrtlabs/mantlemint

```

Copy git clone https://github.com/scrtlabs/mantlemint.git

```

b. If you haven't already,install golang

c. Rungo build -mod=readonly -o build/mantlemint ./sync.go

1b. Run from Binary

If you don't want to compile yourself, you can just download themantlemint executable built for Ubuntu 20.04

```

Copy wget-Omantleminthttps://github.com/scrtlabs/mantlemint/releases/download/v0.0.1-scrt/mantlemint

```

1. Install SGX

Install SGX the same way you would for a node, as described in the Node Setup section

1. Download and unpack thesecretd package

```

Copy
wget"https://github.com/scrtlabs/SecretNetwork/releases/download/v1.3.1/secretnetwork_1.3.1_mainnet_goleveldb_amd64.deb"
sudoaptinstall-y./secretnetwork_1.3.1_mainnet_*_amd64.deb

```

1. Register the node

To allow Mantlemint to sync blocks and run queries that contain encrypted data, we will need to register it.

```

Copy exportSCRT_ENCLAVE_DIR=/usr/lib exportSCRT_SGX_STORAGE=/opt/secret/.sgx_secrets secretdauto-register

```

1. Create a directory for mantlemint data

This directory should be separate from other mantlemint instances or secretd instances

```

Copy mkdir-phome/.mantlemint

```

1. Create an app.toml file

Create aconfig/app.toml file in your mantlemint directory, and set it with a similar app.toml file to what asecretd node uses

```

Copy mkdir-phome/.mantlemint/config

```

Exampleapp.toml file:

```

Copy

# This is a TOML config file.

# For more information, see https://github.com/toml-lang/toml

**Base Configuration**

# The minimum gas prices a validator is willing to accept for processing a
```

transaction. A transaction's fees must meet the minimum of any denomination

specified in this config (e.g. 0.25token1;0.0001token2).

minimum-gas-prices="0.0125uscrt"

default: the last 100 states are kept in addition to every 500th state; pruning at 10 block intervals

nothing: all historic states will be saved, nothing will be deleted (i.e. archiving node)

everything: all saved states will be deleted, storing only the current and previous state; pruning at 10 block intervals

custom: allow pruning options to be manually specified through 'pruning-keep-recent', 'pruning-keep-every', and 'pruning-interval'

pruning="default"

These are applied if and only if the pruning strategy is custom.

pruning-keep-recent="0" pruning-keep-every="50" pruning-interval="0"

HaltHeight contains a non-zero block height at which a node will gracefully

halt and shutdown that can be used to assist upgrades and testing.

Note: Commitment of state will be attempted on the corresponding block.

halt-height=3343000

HaltTime contains a non-zero minimum block time (in Unix seconds) at which

a node will gracefully halt and shutdown that can be used

to assist upgrades

and testing.

Note: Commitment of state will be attempted on the corresponding block.

halt-time=0

MinRetainBlocks defines the minimum block height offset from the current

block being committed, such that all blocks past this offset are pruned

from Tendermint. It is used as part of the process of determining the

ResponseCommit.RetainHeight value during ABCI Commit. A value of 0 indicates

that no blocks should be pruned.

This configuration value is only responsible for pruning Tendermint blocks.

It has no bearing on application state pruning which is determined by the

"pruning-*" configurations.

Note: Tendermint block pruning is dependant on this parameter in conunction

with the unbonding (safety threshold) period, state pruning and state sync

snapshot parameters to determine the correct minimum value of

ResponseCommit.RetainHeight.

min-retain-blocks=0

InterBlockCache enables inter-block caching.

inter-block-cache=true

IndexEvents defines the set of events in the form {eventType}.{attributeKey},

which informs Tendermint what to index. If empty, all events will be indexed.

Example:

["message.sender", "message.recipient"]

index-events=[]

IavlCacheSize set the size of the iavl tree cache.

Default cache size is 50mb.

iavl-cache-size=781250

Telemetry Configuration

[telemetry]

Prefixed with keys to separate services.

service-name=""

Enabled enables the application telemetry functionality. When enabled,

an in-memory sink is also enabled by default. Operators may also enabled

other sinks such as Prometheus.

enabled=false

Enable prefixing gauge values with hostname.

enable-hostname=false

Enable adding hostname to labels.

enable-hostname-label=false

Enable adding service to labels.

enable-service-label=false

PrometheusRetentionTime, when positive, enables a Prometheus metrics sink.

prometheus-retention-time=0

GlobalLabels defines a global set of name/value label tuples applied to all

metrics emitted using the wrapper functions defined in telemetry package.

Example:

[["chain_id", "cosmoshub-1"]]

global-labels=[ ]

API Configuration

[api]

Enable defines if the API server should be enabled.

enable=true

Swagger defines if swagger documentation should automatically be registered.

swagger=true

Address defines the API server to listen on.

address="tcp://0.0.0.0:1317"

MaxOpenConnections defines the number of maximum open connections.

max-open-connections=1000

RPCReadTimeout defines the Tendermint RPC read timeout (in seconds).

rpc-read-timeout=10

RPCWriteTimeout defines the Tendermint RPC write timeout (in seconds).

rpc-write-timeout=0

RPCMaxBodyBytes defines the Tendermint maximum response body (in bytes).

rpc-max-body-bytes=1000000

EnableUnsafeCORS defines if CORS should be enabled (unsafe - use it at your own risk).

enabled-unsafe-cors=true

Rosetta Configuration

[rosetta]

Enable defines if the Rosetta API server should be enabled.

enable=false

Address defines the Rosetta API server to listen on.

address=":8080"

Network defines the name of the blockchain that will be returned by Rosetta.

blockchain="app"

Network defines the name of the network that will be returned by Rosetta.

network="network"

Retries defines the number of retries when connecting to the node before failing.

retries=3

Offline defines if Rosetta server should run in offline mode.

offline=false

gRPC Configuration

[grpc]

Enable defines if the gRPC server should be enabled.

enable=true

Address defines the gRPC server address to bind to.

address="0.0.0.0:9090"

gRPC Web Configuration

[grpc-web]

GRPCWebEnable defines if the gRPC-web should be enabled.

NOTE: gRPC must also be enabled, otherwise, this configuration is a no-op.

enable=true

Address defines the gRPC-web server address to bind to.

address="0.0.0.0:9091"

EnableUnsafeCORS defines if CORS should be enabled (unsafe - use it at your own risk).

enable-unsafe-cors=false

State Sync Configuration

State sync snapshots allow other nodes to rapidly join the network without replaying historical

blocks, instead downloading and applying a snapshot of the application state at a given height.

[state-sync]

snapshot-interval specifies the block interval at which local state sync snapshots are

taken (0 to disable). Must be a multiple of pruning-keep-every.

snapshot-interval=100

snapshot-keep-recent specifies the number of recent snapshots to keep and serve (0 to keep all).

snapshot-keep-recent=2

[wasm]

The maximum gas amount can be spent for contract query.

The contract query will invoke contract execution vm,

so we need to restrict the max usage to prevent DoS attack

contract-query-gas-limit="10000000"

The WASM VM memory cache size in MiB not bytes

contract-memory-cache-size="0"

The WASM VM memory cache size in number of cached modules. Can safely go up to 15, but not recommended for validators

contract-memory-enclave-cache-size="0"

```

1. Download Genesis File

```

Copy wget -O home/.mantlemint/config/genesis.json "https://github.com/scrtlabs/SecretNetwork/releases/download/v1.2.0/genesis.json"

```

1. Download a snapshot

To start mantlemint, we highly recommend using a snapshot. The earlier one starting from block 3340000. This will save having to switch out mantlemint binaries to account for network upgrades.

To download and unpack a snapshot, use the following command

```

Copy wget-qO-https://engfilestorage.blob.core.windows.net/mm-snapshots/height_3340000.tar|tar-xvf--C/path/to/mm/directory

```

Make sure the files are unpacked in your chosen mantlemint directory

Pro Tip: Currently Mantlemint is fully archival. That means snapshots are really large. This command will download and unpack the snapshot without having to use twice the storage amount

1. Finally, you run Mantlemint

Now we are ready to run Mantlemint. It's slightly awkward to run as you have to set multiple environment variables, but they're fairly straightforward. An example run command would be -

```

Copy

Tell SGX we are running on a production chain

SGX_MODE=HW \

Location of genesis file

GENESIS_PATH=home/.mantlemint/config/genesis.json \

Home directory for mantlemint.

Mantlemint will use this to:

- read HOME/config/app.toml

- create and maintain HOME/mantlemint.db directory

- create and maintain HOME/data/* for wasm blobs; (unsafe to share with RPC!)

- create and maintain HOME(INDEXER_DB).db for mantle indexers

MANTLEMINT_HOME=./hd/mm \

Chain ID

CHAIN_ID=secret-4 \

RPC Endpoint; used to sync previous blocks when mantlemint is catching up

RPC_ENDPOINTS=http://20.104.20.173:26657,http://20.104.20.173:26657 \

Name of mantlemint.db, akin to application.db in standard cosmos/tendermint

MANTLEMINT_DB=mantlemint \

Name of indexer db

INDEXER_DB=indexer \

WS Endpoint; used to sync live block as soon as they are available through RPC websocket

WS_ENDPOINTS=ws://20.104.20.173:26657/websocket,ws://20.104.20.173:26657/websocket \

Flag to enable/disable mantlemint sync, mainly for debugging

DISABLE_SYNC=false \

Flags that help mantlemint find the secret-specific libraries and secret keys

SCRT_SGX_STORAGE="/opt/secret/.sgx_secrets" SCRT_ENCLAVE_DIR="/usr/lib" \

./mantlemint --x-crisis-skip-assert-invariants

```

That's it!

Last updated11 months ago On this page *What is Mantlemint? * Features * Prerequisites * Minimum Requirements * Setup

Was this helpful? Edit on GitHub Export as PDF
```