Supra RPC Operator Manual

Introduction 2
Prerequisites 2
Step 1: Setup/Update the Docker container 3
Step 2: Adjust config.toml 4
Step 3: Start the RPC node 4
Step 4: Status Check 5
Step 5: Test the APIs 7

Introduction

Welcome to Supra! This guide will walk you through the steps required to set up an RPC node on our blockchain network.

Prerequisites

- Install Docker Engine or Docker Desktop on your machine using the following https://docs.docker.com/engine/install/.
- You might need to follow the post-installation steps (https://docs.docker.com/engine/install/linux-postinstall/#manage-docker-as-a-non-root-user) too.
- Install Google Cloud CLI (gcloud CLI) using this https://cloud.google.com/sdk/docs/install#linux.
- Install wget, shasum, and jq.
- Please open the below ports for the respective network.
    - Testnet: 26000
    - Mainnet: 30000
- Run the following commands to authorize Gcloud and Docker to pull the image from the image repository:
    - gcloud auth login
    - gcloud auth configure-docker asia-docker.pkg.dev
- Hardware Requirements:
    - Cores: 32 or more
    - RAM: 64G
    - CPU: Intel Xeon (or) AMD EPYC CPU @ 3.2GHz speed or higher
    - Architecture: x86/64
    - Disk Type: SSD
    - Minimum Disk Size: 2TB
    - Network Bandwidth: 2Gbps or more

Step 1: Setup/Update the Docker container

Setup a New RPC node

Run the below commands in a new command line to retrieve the installation script and print its usage information.

None

```
wget -O setup_rpc.sh https://mainnet-data.supra.com/scripts/setup_rpc.sh
chmod +x setup_rpc.sh
./setup_rpc.sh setup
```

You should see the following output after running the last command:

```
Usage: ./setup_rpc.sh setup <image_version> <container_name> <host_supra_home> <network> <validator_ip>
Parameters:
  - image_version: The RPC node Docker image version to use. Must be a valid semantic versioning identifier: i.e. 'v<major>.<minor>.<patch>'.
  - container_name: The name of your Supra RPC Docker container.
  - host_supra_home: The directory on the local host to be mounted as $SUPRA_HOME in the Docker container.
  - network: The network to sync with. Either 'testnet' or 'mainnet'.
  - validator_ip: The IP address of the validator to sync consensus data from.
```

Set each argument in alignment with the usage information and according to your requirements. Use the *<image_version>* provided to you. *<validator_ip>* must be set to the IP address of your validator node.

Your final command should look something like this:

None

```
./setup_rpc.sh setup v8.0.2 supra-testnet-rpc supra_home testnet 0.0.0.0
```

Finally, run the command. After the script completes, run it once more. If you see that the script syncs new data, then the remote snapshot may be undergoing an update. Our snapshots are updated via diff-syncs, so the update should finish quickly. Try running the script again in a few minutes, and continue to do this until the below output shows

that no new data was transferred.

```
Transferred:      346.295 MiB / 72.299 GiB, 0%, 13.072 MiB/s, ETA 1h33m57s
Transferred:          158 / 5002, 3%
Elapsed time:        34.0s
Transferring:
 *                                  000855.sst: 36% /8.155Mi, 0/s, -
 *                                  000857.sst: transferring
 *                                  000858.sst: transferring
 *                                  000859.sst: transferring
```

To Update Your RPC Node

Please follow the instructions to update the version in your RPC node using the same script.

```
None
./setup_rpc.sh update
```

You should see the following output:

```
Usage: ./setup_rpc.sh update <image_version> <container_name> <host_supra_home> <network>
Parameters:
  - image_version: The RPC node Docker image version to use. Must be a valid semantic versioning identifier: i.e. 'v<major>.<minor>.<patch>'.
  - container_name: The name of your Supra RPC Docker container.
  - host_supra_home: The directory on the local host to be mounted as $SUPRA_HOME in the Docker container.
  - network: The network to sync with. Either 'testnet' or 'mainnet'.
```

Set each argument in alignment with the usage information and according to your requirements. Use the *<image_version>* provided to you.

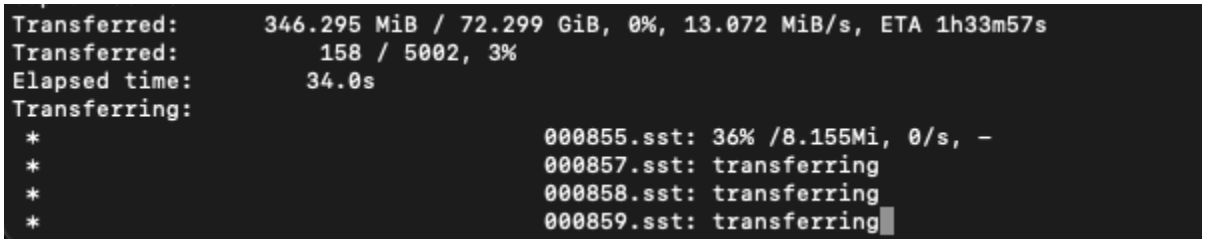
- Testnet Image: v8.0.2
- Mainnet Image: v7.1.8

Your final command should look something like this:

```
None
./setup_rpc.sh update v8.0.2 supra-testnet-rpc supra_home testnet
```

Finally, run the command. After the script completes, run it once more. If you see that the script syncs new data, then the remote snapshot may be undergoing an update. Our snapshots are updated via diff-syncs, so the update should finish quickly. Try running the script again in a few minutes, and continue to do this until the below output shows

that no new data was transferred.



Step 2: Adjust config.toml (Setup Only)

If the script completed successfully, then a file called *config.toml* should have been created inside of the directory that you specified as *<host_supra_home>*. Open this file and edit the contents under the *NODE PARAMETERS* section to suit your requirements. Please note that paths are set relative to the value of *$SUPRA_HOME* in the Docker container. You may wish to modify the list of allowed origins but should not need to change most other parameters.

Step 3: Start the RPC node

After making any required adjustments to *config.toml*, update the copy stored in the Docker image, and start the RPC node by running the below commands.

None

```
docker start <container_name>
docker cp <host_supra_home>/config.toml <container_name>:/supra/
docker exec -itd <container_name> /supra/rpc_node start
```

Step 4: Status Check

Please make sure the following keywords are present in the logs. You can search the logs using a text editor or a command line tool. For example:

Block production check:

None

```
# From within the Docker container:
$ tail -f /supra/configs/rpc_logs/rpc_node.log | grep "Block height"


# From the local host:
$ tail -f <host_supra_home>/rpc_logs/rpc_node.log | grep "Block height"


# Exmaple Output:
[2024-08-18T15:45:58.676917Z+00:00]  INFO execution: Block hash:
(8d555eafc61327f890f94c2e60be0734e41dcde0d682174391546565faa6ce31), Block
height: (134183), Block round: (1918), Block epoch: (61)
[2024-08-18T15:46:01.225994Z+00:00]  INFO execution: Block hash:
(ba05b51d2753f167786ece73d0beef42cfb844474ff47d74b3dc9f83a4fa2968), Block
height: (134184), Block round: (1919), Block epoch: (61)
[2024-08-18T15:46:03.736468Z+00:00]  INFO execution: Block hash:
(86ddad036dafb2e1fd737ee15119df6802ce167b60cf13051e372cababf1eb64), Block
height: (134185), Block round: (1920), Block epoch: (61)
[2024-08-18T15:46:06.247863Z+00:00]  INFO execution: Block hash:
(63310e1418c3a1603f7669384538ac7a7dd6d4079c3e874e20134437ce366428), Block
height: (134186), Block round: (1921), Block epoch: (61)
[2024-08-18T15:46:13.794928Z+00:00]  INFO execution: Block hash:
(7ac7ebbc380cb7c8df771b387df19d72fc4f5e89129a453161fea43b0091d857), Block
height: (134187), Block round: (1923), Block epoch: (61)
[2024-08-18T15:46:16.307156Z+00:00]  INFO execution: Block hash:
(75de064e9ec85cce70387ad0a2a41ede5767131412df32ff95e48f0f20f939b1), Block
height: (134188), Block round: (1924), Block epoch: (61)
[2024-08-18T15:46:18.819560Z+00:00]  INFO execution: Block hash:
(cfe1b1c6bc594ed313cbca03280b8099c016dc45d817a0dc62bd499367e5b484), Block
height: (134189), Block round: (1925), Block epoch: (61)
[2024-08-18T15:46:21.332839Z+00:00]  INFO execution: Block hash:
(3ef98386c923719afed2c5f6c9e3815c0777c492dbc30338bd44a4672d890743), Block
height: (134190), Block round: (1926), Block epoch: (61)
[2024-08-18T15:46:23.844275Z+00:00]  INFO execution: Block hash:
(7e74b62e2207de9daf3bb696c0b0cf9725b2a6ec4e4340a4e6aa2b96e3b7099f), Block
height: (134191), Block round: (1927), Block epoch: (61)
```

This indicates that your RPC node has caught up with its attached validator and is now in sync with the network.

Step 5: Test the APIs

The below script provides some simple *curl* commands for testing the REST API endpoints. Notice that the faucet endpoint uses the load-balancer for the network while the others refer to the local host. This is because your RPC node is not configured to run in faucet mode. **Please note that the faucet is only available in testnet: You will not be able to test this functionality in mainnet.**

- Replace <RPC Port> in the below code snippet according to the value set for the *bind_addr* in the *config.toml* for the environment that you want to test. If you have made no changes to the files provided in *Step 2*, then use the following.
    - Testnet: `26000`
    - Mainnet: `30000`

None

```
#!/bin/bash

RPC_PORT=<RPC Port>

############################ Faucet ############################



curl -i --request GET \

      --url
"https://rpc-testnet.supra.com/rpc/v1/wallet/faucet/1bd097c16fc9556f7f1038e18400260a
c3150bc7499ca6bacfb9c840cbcb21f6" \

      --header "Accept: application/json"

#

# E.g. output: 0x231f998d963dc736831ae3b4adfbd10b792fd7ae4278b3010cb81c265ceae1f0
```

```
########################### Transactions ############################



curl -i --request GET \

       --url "http://localhost:${RPC_PORT}/rpc/v1/transactions/chain_id" \

       --header "Accept: application/json"



# curl -i --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/transactions/0x231f998d963dc736831ae3b4adfbd10b
792fd7ae4278b3010cb81c265ceae1f0" \

#      --header "Accept: application/json"





########################### Blocks ############################



curl -i --request GET \

       --url "http://localhost:${RPC_PORT}/rpc/v1/block" \

       --header "Accept: application/json"



# curl --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/block/height/54?with_finalized_transactions=tru
e" \

#      --header "Accept: application/json"
```

```
# curl --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/block/0xbce007170b1dc8d12ce44b21e0fe8fb2c5049a4
f871b0c346a9dcfab31bd1655" \

#      --header "Accept: application/json"



# curl --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/block/0xbce007170b1dc8d12ce44b21e0fe8fb2c5049a4
f871b0c346a9dcfab31bd1655/transactions" \

#      --header "Accept: application/json"




############################ Accounts ############################



# curl --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/accounts/0x1bd097c16fc9556f7f1038e18400260ac315
0bc7499ca6bacfb9c840cbcb21f6/transactions" \

#      --header "Accept: application/json"



# curl --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/accounts/0x1bd097c16fc9556f7f1038e18400260ac315
0bc7499ca6bacfb9c840cbcb21f6" \
```

```
#      --header "Accept: application/json"



# curl --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/accounts/0x1bd097c16fc9556f7f1038e18400260ac315
0bc7499ca6bacfb9c840cbcb21f6/resources" \

#      --header "Accept: application/json"



# curl --request GET \

#      --url
"http://localhost:${RPC_PORT}/rpc/v1/accounts/0x1bd097c16fc9556f7f1038e18400260ac315
0bc7499ca6bacfb9c840cbcb21f6/resources/0x1::coin::CoinStore%3C0x1::supra_coin::Supra
Coin%3E" \

#      --header "Accept: application/json"
```