Raiden Services Documentation

Release 2.0.1

Brainbot Labs Est.

Oct 27, 2022

CONTENTS:

1 Monitoring Service **1**

2 Pathfinding Service **3**

3 Installation **5**

4 Running the services **7**

4.1 Pathfinding Service 8

4.2 Monitoring Service 8

4.3 Claim Pathfinding Fees 9

5 Indices and tables **11**

CHAPTER

ONE

MONITORING SERVICE

The Monitoring Service watches open payment channels when the user is not online. In case one of the user's channel partners wants to close a channel while the user is offline, the monitoring service sends the latest balance proof to the smart contract and thus ensures the correct settlement of the channel.

CHAPTER

TWO

PATHFINDING SERVICE

The Pathfinding service supports users in finding the cheapest or shortest way to route a payment through the network. A pathfinding service relies on data from the network, the respective smart contract as well as information provided voluntarily by mediating nodes. This information consists of the mediation fees charged and the respective available channel capacities.

CHAPTER

THREE

INSTALLATION

The Raiden services are available through the pip package `raiden-services`. As there is currently a lot of development activity, we recommend to use the latest version from git right now.

The Raiden services require Python 3.8+. To install the latest version of the services follow these instructions:

```
git clone git@github.com:raiden-network/raiden-services.git
cd raiden-services
virtualenv -p python3.8 venv
source venv/bin/activate
pip install -U -r requirements-dev.txt
pip install -e .
```

This installs the executables `pathfinding-service`, `monitoring-service` and `request-collector` into the virtual environment.

CHAPTER

FOUR

RUNNING THE SERVICES

The services share a number of options. These are:

`--keystore-file`
Defines the path to the keystore file. (*Required*)

`--log-level`
Defines the minimal level of log messages to be shown.

Defaults to `INFO`.

`--log-json/--no-log-json`
Enables or disables the logging to be JSON formatted.

Defaults to `False`.

`--eth-rpc`
Defines the URI of the Ethereum node to be used.

Defaults to `http://localhost:8545`.

`--token-network-registry-contract-address`
Defines the address of the token network registry to be used.

Defaults to the contract address that the Raiden client uses for the given chain.

`--user-deposit-contract-address`
Defines the address of the user deposit contract to be used.

Defaults to the contract address that Raiden the client uses for the given chain.

`--start-block`
Defines the block number at which the service starts syncing.

Defaults to the block number in which the earliest contract was deployed, but only if predeployed contracts are used. Otherwise it is set to 0.

`--confirmations`
Defines the number of blocks after which blocks are assumed confirmed.

Defaults to 8 blocks.

4.1 Pathfinding Service

The Pathfinding service has some additional parameters which can be set.

`--state-db`
Defines the location where the state is stored.

Defaults to `~/.config/raiden-pathfinding-service/state.db` on Unix or `~/Library/Application Support/raiden-pathfinding-service/state.db` on macOS.

`--host`
Defines the URI the REST API of the Pathfinding service will be available at.

Defaults to `localhost`.

`--service-fee`
Defines the minimum service fee a client has to pay when requesting routes.

Defaults to zero.

`--enable-debug`
Enabled the extended REST API for debugging. We strongly advise against enabling this option on nodes that are reachable on the network.

Disabled by default.

4.2 Monitoring Service

The Monitoring service consists of two components that have to be started independently for now. The `monitoring-service` program is responsible for all interactions with the blockchain and **must** be started first, as it writes some essential information to the state database. Afterwards the `request-collector` can be started. It listens to the public Matrix room for monitoring request from Raiden nodes and validates and saves them to the monitoring services database.

Note: The reason for this separation is security. Even when request collector are attacked, the monitoring service continues to work and can safely monitor the user's channels.

The Monitoring service has some additional parameters which can be set.

`--state-db`
Defines the location where the state is stored.

Defaults to `~/.config/raiden-monitoring-service/state.db` on Unix or `~/Library/Application Support/raiden-monitoring-service/state.db` on macOS.

`--monitor-contract-address`
Defines the address of the monitoring contract to be used.

Defaults to the contract address that the Raiden client uses for the given chain.

`--min-reward`
Defines the minimum reward a client has to pay when sending a request for monitoring.

Defaults to zero.

The request collector shares the database path with the monitoring service

--state-db
Defines the location where the state is of the monitoring service is stored.

Defaults to `~/.config/raiden-monitoring-service/state.db` on Unix or `~/Library/Application Support/raiden-monitoring-service/state.db` on macOS.

4.3 Claim Pathfinding Fees

If you are running the Pathfinding Service with `--service-fee`, the service fees will need to be claimed to actually reach your wallet. Claiming fees less frequently will reduce the Ethereum transactions costs spent while claiming, while not claiming for a long time might make you miss the expiration time for your claims. The `claim-pfs-fees` script helps doing this in a profitable and reasonably easy way. You must use the same `keystore` and `state-db` as for running the Pathfinding Service.

In addition to the mandatory parameters, the following parameters are useful to increase profitability:

--rdn-per-eth
The RDN/ETH price used to calculate whether the value of a fee IOU exceeds the transactions costs to claim it.

--expire-within
Only IOUs which are claimable until the time range of now plus this number of seconds will be claimed. Reducing this number gives the IOUs more time to accumulate a higher amount, allowing higher amounts to be claimed with the same Ethereum transaction costs. Increasing this value makes it less likely for IOUs to expire before being claimed, if you run `claim-pfs-fees` rarely or at irregular intervals.

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- search