The Electrum Server

Last change: 15MAY21, hyperpeek#9099
Tips: RKJLzw6kB6f4nm7Fwuy4aaK4ygSSfw9qCe

Introduction

The RVN electrum wallet – currently one of the few that support the hardware wallets Trezor and Ledger – relies on special servers as an intermediary to the RVN block chain. The server interacts with a full rvn node via the rpc interface to supply (with the help of a database) transaction and payment data to the remote electrum wallets, so there is no need for the wallets to keep a full copy of the block chain, making them more light-weight than full node client wallets. Outgoing transactions are signed on the electrum wallet client (in case of Trezor or Ledger inside the hardware itself) and are transmitted via the electrum server through the rvn nodes into the network, so all sensitive data like private keys are kept in the client (or even hardware), making this a very secure solution. Even though the electrum protocol is optimized for low bandwidth and load, each server can serve only a certain amount of clients and especially with many new users trying syncing a new electrum installation from the genesis block, a single electrum server can be stretched beyond its limits pretty fast. This as well as general security and backup concerns call for a more distributed electrum server infrastructure. The aim of the guide is therefore to give a step-by-step instruction about setting up a RVN electrum server running beside a full RVN node client.

System requirements

While pretty much any Linux distribution should work, this guide uses Ubuntu 20.04 running on a VPS server as an example setup. Performance-wise, minimum specs are 40 GB of disk space, 2 GB RAM + 2 GB swap and preferably unlimited network traffic, as 1 TB can be easily reached within a few days. CPU is not too much of a concern, as the electrumx server we will use is mostly implemented in Python running in a single thread. Higher CPU spec- and / or count may speed up syncing, though a reliable and fast network connection is still the best guarantee for a seamless operation of the system. The following guide assumes you are running the raven and electrumx server as a non-root user and all commands should be run from the user account created.

Step-by-step setup of the node

- Setup a system running Ubuntu 20.04 or install it through a pre-build image with any of the countless VPS providers. After installation, it is good practice to do a sudo apt update to get the installation sources provided by your VPS host and bring the system up-to-date with sudo apt upgrade.

- Setup a user account (e.g. electrumx). You may want to setup SSH key authentication with this account and remove root access (once ssh with key is proved working) in /etc/ssh/sshd_config, so any subsequent login can only be through this account putting an extra layer of security.

- Check https://github.com/RavenProject/Ravencoin/releases for the latest release of raven-x.x.x.x-x86_64-linux-gnu.zip. Download (e.g. through **wget**) and unpack the archive inside the zip – this will probably require installing unzip through **sudo apt install unzip**.

- Create a directory .raven (**mkdir .raven**) in your home directory (e.g., /home/electrumx) and create a file raven.conf inside this directory with the following content (vi, vim, nano etc.):

```
rpcuser=<some_username>
rpcpassword=<some_password>
rpcallowip=127.0.0.1
listen=1
server=1
daemon=1
txindex=1
assetindex=1
maxconnections=64
```

- Note the txindex=1 and assetindex=1 above, which is required for the electrumx server. If you already run a node without this setting, add it and perform a restart of ravend with the option –reindex. This will take significant time.

- For better maintenance during updates, you may want to set up a link to your raven binary directory, e.g.:

  ```
  ln –s /home/electrumx/raven-x.x.x.x /home/electrumx/raven
  ```

  This way you can always access the binary files through ./raven/bin/... from your home directory.

- You can now start the raven node with **./raven/bin/ravend** . After some time you can check the state of the node using **./raven/bin/raven-cli getinfo** . Keep an eye on the “blocks” – these will remain at 0 until all headers are synced (can be half an hour depending on the system and connection) and it will ultimately reach the current block-height of the rvn chain. By this time, which is normal to take several hours, about 20 GB of data will have been downloaded into the working ./.raven directory.

- At this point you could install some script or service into /etc/systemd/system to run the node automatically at server startup. This ensures a 24/7 operation in case of an unexpected server restart due to system update or maintenance. You could also install watch-dog scripts to restart the server in case of a crash, however, this is beyond the scope of this document. See https://medium.com/@benmorel/creating-a-linux-service-with-systemd-611b5c8b91d6 for an example.

- Although you could continue with installing the electrumx server, it is suggested that you wait until the Ravencoin chain is fully synced to avoid later issues. You will not be able to start the electrumx server without a fully synced chain so some patience is required here.

Step-by-step setup of the electrumx server

The electrumx server is almost entirely written in python and can be found at https://github.com/Electrum-RVN-SIG/electrumx-ravencoin/releases/.

Full documentation and information about installation / running of the electrumx server can be found at https://electrumx-ravencoin.readthedocs.io/en/latest/. This is also the place to look first if you run into issues with the following guide, which only briefly describes the steps required.

It is assumed that you have basic knowledge of editing files on Linux (vi, vim, nano etc.) and also basic knowledge of Python is helpful in case errors come up during the installation.

- Download and unpack the latest version of electrumx ravencoin from the link above (e.g. wget <path>) – the tar.gz version is suggested as that can easily be unpacked with tar –xzf {name}.tar.gz .

- Make a symbolic link to the electrumx, e.g., **ln -s {name} electrumx** so the version can be found by the scripts in-system.

- Make sure Python 3.8.5 is installed by typing **python3 –version**

- Now install pip3:

- Install general build environment and python pip:
  ```
  sudo apt install python3-pip
  sudo apt install virtualenv
  sudo apt install gcc
  sudo apt install build-essential
  sudo apt install python3-dev
  sudo apt install cmake
  ```

- Copy the config file and startup-script:
  ```
  sudo cp ./electrumx/contrib/systemd/electrumx.service /etc/systemd/system/
  sudo cp ./electrumx/contrib/systemd/electrumx.conf /etc/
  ```

- The service file points to a script that creates a virtual environment and downloads the required python modules for the server.

- Create a directory for database and the ssl certificates and generate a self-signed one:
  ```
  mkdir electrumx_db
  mkdir ssl_cert
  cd ssl_cert
  openssl genrsa -out server.key 2048
  openssl req -new -key server.key -out server.csr
  ```
  (information does not matter, no pwd)
  ```
  openssl x509 -req -days 1825 -in server.csr -signkey server.key -out server.crt
  cd ..
  ```

- Edit the conf file: **sudo vi /etc/electrumx.conf** :
  ```
  DB_DIRECTORY = /home/electrumx/electrumx_db
  DAEMON_URL = http://<rpcuser>:<rpcpwd>@localhost (user/pwd from rpc in raven.conf)
  COIN = Ravencoin
  SERVICES = ssl://:50002
  REPORT_SERVICES = ssl://(your ip/domain here):50002
  SSL_CERTFILE = /home/electrumx/ssl_cert/server.crt
  ```

```
SSL_KEYFILE = /home/electrumx/ssl_cert/server.key
COST_SOFT_LIMIT = 0
COST_HARD_LIMIT = 0
BANDWIDTH_UNIT_COST = 1000
CACHE_MB = {Defaults to 1200 MB in memory. You may decrease this if
this number is too much}
```

- See https://electrumx-ravencoin.readthedocs.io/en/latest/environment.html for a full list of environment variables. You should at least briefly look over this too see if you need to change anything for your server

- Now double-check that the rvn node is running and up-to-date with the blocks:**./raven/bin/raven-cli getinfo** if that looks ok you can start the electrumx server: **systemctl start electrumx**
  And check the progress and log of the server: **journalctl -u electrumx –f**
  Stopping the server with: **systemctl stop electrumx**

The electrum server tends to eat up memory and CPU during the initial syncing. This is not a problem once the server has been synced.

Once the electrumx database is fully synced (takes about 3-4h) the server will start servicing connections on port 50002, so you can try and connect your electrum-rvn wallet. Be aware that there can be significant data going out of the server, so make sure it is on an unlimited or large-enough data-plan or you regularly check the data usage.

In case the server hangs with a message like “electrumx_server[490]: struct.error: 'H' format requires 0 <= number <= 65535” or if you are running low on disk space, you can try to stop the server and compact the electrumx server’s history using the commands:

```
systemctl stop electrumx
export COIN=Ravencoin
export DB_DIRECTORY=/home/electrumx/electrumx_db
./electrumx/electrumx_compact_history
systemctl start electrumx
```

After about 10-15 minutes the database should be compacted and fixed and you can restart the server.