duality
blockchain solutions

Tutorial: How to build a Sequence (SEQ) staker on a Raspberry Pi

Summary

This tutorial should help you set-up your Raspberry Pi as a 24/7 low power staking device for Sequence (SEQ). You will require a Unix-based system (any Linux like Ubuntu or MacOS) or a dedicated ssh-client (like putty) to login to your Raspberry.

Preparing the Raspberry Pi

- Get the latest raspbian-lite image from the Raspbian download page
- Flash SD card according to the official intructions
- Insert SD card into your Raspberry Pi, connect LAN cable, connect power supply
- Find IP address of raspberry
- `ssh` into your raspberry

  ```
  ssh pi@<YOUR LAN IP ADDRESS>
  ```

  The standard user is "pi" with password "raspberry"
- Configure your raspbery by running `sudo raspi-config`:
  - Expand filesystem
  - Change password for the user pi
  - Overclock to your likings (useful for compilation, see below, but not really required for running the very slick Sequence client)
  - Under "Advanced options", you can set the shared memory for the GPU to only 16MB. We won't be running any graphical applications in any case.
- Edit swap file size to 1024MB (see this guide). We use nano as the standard editor. To write the output use `<CTRL>+o` and to exit `<CTRL>+x`

  ```
  sudo nano /etc/dphys-swapfile
  ```

  and change the default value of `CONF_SWAPSIZE=100` to `CONF_SWAPSIZE=1024`
- Reboot the pi

Installing the pre-requisites

- After reboot, login again and install the basic dependencies needed for Sequence. All details are on the Sequence github page
- Refresh the list of repositories, and upgrade (you can get a cup of tea in the meanwhile)

  ```
  sudo apt-get update
  sudo apt-get upgrade
  sudo apt-get dist-upgrade
  ```

- Install the basic packages

```
sudo apt-get install build-essential libtool autotools-dev autoconf pkg-config libssl-dev
    libcrypto++-dev libdb++-dev libboost-all-dev
```

- optional: `sudo apt-get install miniupnpc`
- Install git

```
sudo apt-get install git
```

- Set root password

```
sudo passwd root
```

Getting the Sequence binaries

There is two ways to obtain binary (executable) files for Sequence. The easiest way is to download them from the homepage. If you want to be more bleeding edge (or if you are an experienced user), you might want to build from source.

(a) Downloading and using the binaries

- First, we need to download the archive containing the Sequence ARM binaries and extract it. We use a shortened link here, but this is just a redirect to our Sequence ARM release on github

```
wget https://goo.gl/wqMgVW
tar -xzf Sequence-ARMHF-x86-v1.0.0.0.tar.gz
```

- Change into the sequence binary directory to see the binaries

```
cd sequence-1.0.0/bin
ls -lisa
```

  This should show you the binaries. In the following, we will need only the daemon `sequenced` and the client `sequence-cli`

- In order to start the binaries from any directory in the system, you need to add this directory to your `$PATH` variable or link them to a place where the system can find it. If you want to do the latter, I suggest linking them to `/usr/local/bin`:

```
sudo ln -s $(pwd)/sequenced /usr/local/bin/sequenced
sudo ln -s $(pwd)/sequence-cli /usr/local/bin/sequence-cli
```

- Now you should be able to run `sequenced` and `sequence-cli` from anywhere

(b) Building from source

Build Berkeley DB 4.8

Unfortunately, for there are no packages available for DB 4.8 for Raspbian. So we have to build it ourselves.

- Let's assume that you are in your home directory ~/
- Pick some path to install Berkeley DB4.8 (DBD). Here, we just install it a folder called "BDB4.8"

```
mkdir BDB4.8
cd BDB4.8
```

- Now that we are in the directory where we will build BDB. Let us define an environement variable that contains that folder to use later

```
BDB_PREFIX=$(pwd)
```

- Fetch the source and verify that it is not tampered with

```
wget 'http://download.oracle.com/berkeley-db/db-4.8.30.NC.tar.gz'
echo '12edc0df75bf9abd7f82f821795bcee50f42cb2e5f76a6a281b85732798364ef db-4.8.30.NC.tar.gz' |
    sha256sum -c
tar -xzvf db-4.8.30.NC.tar.gz
```

- Build the library and install to our prefix (get a small cup of tea)

```
cd db-4.8.30.NC/build_unix/
../dist/configure --prefix=$BDB_PREFIX --enable-cxx --with-pic
```

- Build and install

```
make
sudo make install
```

Build Sequence

- Clone Sequence using `git`:

```
git clone https://github.com/duality-solutions/Sequence
```

  and enter the directory

```
cd Sequence
```

- Generate installation files:

```
./autogen.sh
```

- Configure installation:

```
./configure --without-gui --disable-tests LDFLAGS="-L${BDB_PREFIX}/lib/"
    CPPFLAGS="-I${BDB_PREFIX}/include/"
```

- Compile and install

```
make
sudo make install
```

Running a Sequence node

- Start the Sequence daemon `sequenced`. This will be always running in the background, and you will interact with it via sending rpc commands from using `sequence-cli`. Running this for the first time will generate the hidden folder `~/.sequence` and a the configuration file `~/.sequence/sequence.conf`
- Try starting the daemon:

```
sequenced
```

- If you start a new installation, the Sequence Daemon might complain about not having a correct RPC user and password. In that case press `CRTL+c` to kill the daemon.
- Start the daemon again in daemon (background) mode:

```
sequenced --daemon
```

Interacting with the Daemon

- Now you are ready to interact with the daemon using `sequence-cli`
- For an overview of commands use `sequence-cli` **`help`**
- Check the status

  ```
  sequence-cli getinfo
  ```

- If you see an error ''wallet loading'' then don't worry: you were simply too fast and the wallet file is still building. Give it two minutes and try again
- At this point, you should wait until Sequence has synchronized the chain. Syncing can take a lot of time. You will recognize that the chain is fully synced, if the block number increases more or less every minute by 1.

Staking

We assume that you have a main machine with the Sequence QT wallet running with a wallet containing funds that you want to have staking. In general, to dump keys or the entire wallet, your wallet needs to be unlocked if it is encrypted.

(a) Staking with a single address

- On your main machine, create a new address that you want to use for staking. If you want, use the splitcoin feature to generate this address already with a multitude of inputs.
- Now go to Tools/Debug Console and type `dumpprivkey <address>` and note the output as `<privkey>`
- On the Raspberry PI, then use `sequence-cli importprivkey <privkey>`
- Repeat for multiple addresses

(b) Staking with the entire wallet

- On the main machine, go to Tools/Debug Console and type `dumpwallet "<filename>"`. If you enter `"<filename>"` without an absolut path, it will be created in the current working directory of the QT-wallet. Notice that the file created contains all your private keys in an unencrypted form. Don't give this file to anyone or leave it lying around!
- Copy the wallet to your Raspberry Pi using `scp` or `putty`
- Go to your Raspberry and import the wallet dump file generated by typing `sequence-cli importwallet "<filename>"`
- In order to make sure that no conflict arises between different clients on the same IP, you should disable staking in your QT wallet. If you don't, the staker might block at some point. In order to disable staking in the QT wallet go to Tools/Open Configuration File and add the line `stakegen=0`.

Congratulations

You have done it! Happy staking. :)