FairChains Manual

How to set-up a

Proof-of-Cooperation Blockchain

Document version 1.1

FairChains Code Development: https://github.com/FairChains
Proof-of-Cooperation & FairCoin White Paper: https://fair-coin.org/white-paper

August, 2018

by Thomas König, Roland Alton & Sebastian Kuehs
Contact: support@fair-coin.org

How to set-up FairChains

Create your own proof-of-cooperation private or public blockchain.
Here we show the first steps for your own chain with one node that creates the blocks called CVN (Collaboratively Validated Node).

This tutorial assumes that you’re using the 64bit version of the operating system Ubuntu 16.04 or Debian Stretch.

Preparation

1. Download the FairChains tarball and extract it.
   https://download.faircoin.world/core/fairchains-lastest-linux-x86_64.tar.bz2

2. Now extract archive:
   ```
   tar xvf fairchains-lastest-linux-x86_64.tar.bz2
   ```

Design your blockchain

The FairChains-Tool is used to create a JSON-file which will be read by the FairChains wallet and contains all required information for the new blockchain. This way it’s not required to compile a new wallet version for each blockchain, all parameters can be configured.

You will be prompted to enter some values which describe your blockchain. A default value will be displayed in brackets mostly. To accept that default value simply press the *enter key.*

Below we provide a walk-through example for a new blockchain. Its name is testchain and it’ll contain TestCoins.

1. Start the tool to create the blockchain specific JSON-file like this:
   ```
   ./fairchains-tool
   ```
2. **Enter a password**
   It must consist of at least 10 characters
3. **Chain name**:
   ```
   testchain
   ```
4. **Currency name:**
   ```
   TestCoin
   ```
5. **Currency symbol:**
   ```
   TEST
   ```

6. **Network magic bytes (0xfabfb5fa)**: We recommend to change the last byte with random value like this:

   0xfabfb5**c3**

7. Maximum amount of money that this blockchain can handle. No more than this amount can be created.

   Maximum amount of coins (money supply) in the blockchain:

   1000000

8. **Network TCP port (49404)**: *Enter the peer-to-peer network TCP port of your choice*

9. For a wallet to receive IP addresses of network peers, it first connects to the DNS server to resolve the seed node names. This will return one or more IP addresses the new node can connect to. In the next step, enter the host name that resolves to hosts that run the new blockchain.

   Seed nodes (One per line. End input by entering '.' + enter): *Enter your DNS seed nodes. e.g. testchain-seed1.example.com* (this example will not work for your chain)

10. In case a client can not resolve the DNS name or you didn't provide a valid one, the client wallet will then try to connect the IP addresses provided in the next step.

    IPv4 and/or IPv6 addresses of fixed seed nodes (One per line. End input by entering '.' + enter)

11. **Public key address version (95)**: Use the version number of your choice to start your address with a certain character. For a list of version to character mapping see here: https://en.bitcoin.it/wiki/List_of_address_prefixes

    In this example we want the addresses to start with a lower case 't'.

    127

12. **Script address version (36)**: See above, but for multi-signature addresses

    In this example we want the addresses to start with an upper case 'T'.

    65

13. **Secret key version (223)**: See above, but for WIF-key (wallet import format)

14. **Extended public key prefix (0x0488b21e)**: Only for advanced users, if you're not sure just press *enter.*

15. **Extended secret key prefix (0x0488ade4)**: See above.

16. **Require standard transactions (true)**: *For standard transactions confirm with enter.* Most users will set this to true.

17. **Blockchain start unix timestamp**: Enter the desired start time of the block chain as the *unix timestamp.* Default is 'now'.

18. **Id of the genesis CVN (0xc0ff0001)**: Most users will accept *default.*

19. **Id of the genesis chain admin (0xadff0001):** Most users will accept *default*.
20. **Block spacing time - in seconds (180)**: *Enter block spacing time.* This represents the time between two successive blocks.
21. **Block spacing grace period time - in seconds (60)**: If a CVN fails to create its block, all other CVNs wait for this time before they choose the next CVN. Most users will accept the *default*.
22. **Transaction fee (0):** The fee is expressed in the unit of Satoshi = multiply the desired value by 100 000 000. FairCoin has currently a fee of 0,008 FAIR × 100 000 000 = 800 000 Satoshi

    The initial value should be set to zero and will be changed later in the blockchain by a dynamic chain parameter payload created by the chain admins.
23. **Dust threshold (0)**: The value defined by the network as the smallest transferable value. This is usually the same, but not lower than the transaction fee.

    The initial value should be set to zero and will be changed later in the blockchain by a dynamic chain parameter payload created by the chain admins.
24. **Maximum block size (1500000)**: Block size in bytes. Block size can be lower for blockchains with an expected small number of transactions. It’s save to accept the default.
25. **Block propagation wait time (50)**: Should correspond with block target. Waiting time in seconds of CVNs after block creation for next actions. Time must be about 60 percent of the block target time.
26. **Retry new signature set interval (15)**: If a CVN fails to create signatures, so that next CVN can create its block, then creation of new chain signatures will be performed every x seconds (e.g. 15 seconds)
27. **Coinbase maturity - in blocks (10)**: Enter the number of blocks coin supply and transaction fees need to mature.

Congrats! The parameters for your chain, three certificates and one JSON-file were generated:

0xadff0001.pem   (Admin certificate)
0xc0ff0001.pem (CVN certificate)
alert-testchain.pem (network alert signing certificate)
testchain.json (chain parameters)

Now distribute the *testchain.json* file to each participant in the network. It contains all the required information to connect to your new blockchain.

The initial start of your blockchain

The testchain.json must be placed in the data directory of the wallet. (default: ~/.fairchains). Move the certificates to the testchain subdirectory. In our example: ~/.fairchains/testchain

1. In a data directory, create a *fairchains.conf* file with the following content:

```
netname=testchain
txindex=1
# is a CVN, so define as a generator
gen=1
# secure with file or fasito (= hardware device to generate keys)
cvn=file
cvnkeyfile=0xc0ff0001.pem
cvncertfile=0xc0ff0001.pem
```

2. Start your CVN, do not connect to other CVNs and do not wait for peers in this test scenario

```
./fairchains-qt -datadir=data -connect=0 -cvnwaitforpeers=0 -printtoconsole
```

   You'll get the message 'Block chain download in progress. Waiting...` If the last block time lies more then 60 min. in the past, you have to append the -maxtipage parameter to the command like this:

```
./fairchains-qt -datadir=data -connect=0 -cvnwaitforpeers=0 -printtoconsole -maxtipage=9999999
```

3. When asked for a password, *enter the password* that was created using the FairChains-Tool.
4. The JSON-file can be signed by the developer team which makes it an 'official chain'. In the wallet you'll see the message: This is an official chain!

   This system is used to prevent fraud if someone creates fake duplicate of your blockchain.
5. Open the Debug Console from the Help menu and start a chain admin session by typing the following into the wallet console:

```
fasitologin file fairchainspass \"0xadff0001.pem\"
```

6. Choose an address of your wallet where you want to receive the initial coin supply. (From the File menu → Receiving addresses...) Add coin supply parameter for adding coins. Use "true" for a final coin supply, or "false" if you want to add additional coins later.

7. Now enter the following command sequence in the wallet's debug console:

```
fasitononce

addcoinsupply tbxELeY6Y3TnJN3cceyPmTf1rMe4iMm 1000000 false "initial coin supply
for testchain blockchain test"

fasitosign <hash value from the previous step>

addcoinsupply tbxELeY6Y3TnJN3cceyPmTf1rMe4iMm 1000000 false "initial coin supply
for testchain blockchain test"
```

8. Now you can start to send coins to any node which is connected to your blockchain.

9. Add new CVNs to distribute the block-generation, facilitating the low-energy proof-of-cooperation mechanism. For details see https://fair-coin.org/white-paper