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 you the first steps for your own chain with one CVN (Collaboratvely Validating Node) only.

This tutorial is based on operating system Ubuntu 16.04 or Debian Stretch.

Preparation

1. Download the FairChains tarball and extract.
   https://download.faircoin.world/core/fairchains.tar.bz2

2. Now extract archive:

   ```
   tar xvf fairchains.tar.bz2
   ```

Design your blockchain

3. Start writing the blockchain specific json file with

   ```
   ./fairchains-tool
   ```

   This way we do not need to compile for each blockchain, all parameters can be configured. Most default values can be simply confirmed with *enter.*

4. **Define a password**: (at least 10 characters)

   ```
   fairchainspass
   ```

5. **Define the name of the chain**:

   ```
   testchain
   ```

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

   ```
   0xfabfb5c3
   ```

7. **Network TCP port (49404)**: *Enter port of your choice* or confirm with *enter* the suggested default

8. 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. seed1.fair-coin.org (this example will not work for your chain)

9. If the 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.
   IP4 and IP6 of fixed seed nodes are just for emergency case.

10. **Public key address version (95)**: Use 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

11. **Script address version (36)**: It's the same like above, just for multi-signature

12. **Secret key version (223)**: The same, but for wif-key (wallet import format)

13. **Extended public key prefix (0x0488b21e)**: Only for advanced users, if you don't know just press *enter.*

14. **Extended secret key prefix (0x0488ade4)**: See above point 13

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

16. **Blockchain start unix timestamp (1533566333)**: *Use unix timestamp.* Default is 'now'.

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

18. **Id of the genesis chain admin (0xadff0001):** Most users will accept *default*.

19. **Block spacing time - in seconds (180)**: *Enter block spacing time.* This represents the time between two successive blocks.

20. **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*.

21. **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

22. **Dust threshold (0)**: The value defined by the network as the smallest transformable value. This is usually the same or at least not lower than the transaction fee.

23. **Maximum block size (1500000)**: Block size in bytes. Block size can be lower for blockchains with an expected small number of transactions.

24. **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.

25. **Retry new signature set interval (15)**: If a CVNs fail 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)

26. **Coinbase maturity - in blocks (10)**: Enter the number of blocks coin supply and transaction fees need to mature.

27. 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)

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

29. In a data directory, create a *faircoin.conf* file with:

```
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
```

30. Move all pem cert files to data/testchain

31. 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
```

    As long as you run only one CVN you may need to add the parameter -maxtipage=9999999 when restarting it after shutdown or hibernation. The following must be entered for reactivation:

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

32. When asked for a password, *enter the password* defined above.

33. For the json file a signature can additionally be added by the developer team, to hash the two public keys in the core wallet. With this the wallet can verify whether it is an official chain.

34. Now start a chain admin session by typing the following into the wallet console:

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

35. Add coin supply parameter for adding coins. Use “true” for a final coin supply, or “false” if you want to add additional coins later.

```
fasitononce
addcoinsupply fbxELeY6Y3TnJN3cceyPmTf1rMe4iMm6V2 1000000 false "initial coin supply for testchain blockchain test"
fasitosign “yourhash”
addcoinsupply fbxELeY6Y3TnJN3cceyPmTf1rMe4iMm6V2 1000000 false "initial coin supply for testchain blockchain test"
```

    With the second coin supply command the coins or tokens are sent to the network.

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

37. 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

38. Via the FairChain node API you can connect Omni Layer to create rules for your assets or simply give access by the web wallet to your users.