Protocol Brief View

The code repository can basically be divided in 2 parts. One network component and one security part. The security part checks that the current blockchain and new blocks are correct (signature, legal transaction pow etc). It can also dowload a blockchain in cache from another peer, check if it is correct and if so replace its own blockchain.

The network part diffuses and update blockchain between peers. In order to do that each peer frequently periodically emit hello messages where they provide the number and the last hash of the block. Other peers can record a tables of the states of the peer (delayed, if they have smaller id last block , inconsistent if wrong hash etc.) If they notice there is some better peer than them (i.e. a peer that has a higher last block number) they can first download the hashs of missing blocks and check the pow. Then if it is correct (checking pow avoid flooding by some malicious peer), they can download the blocks and checks all the transaction.

Miner can forge new block with the security part. Some node could also it’s one transaction post transaction signed on the network (not implemented currently).

Diagram state of peers

Hello message received

Delayed

Heard

Reach time-out

Correct hash & Bnum

Deleted

Synchronized

Correct hash but lower

Bnum

Inconsistent

Sign

PeerTables

Database

Check

Security

Network

Input Output & debug Services

Network :

MuxDemux

, HelloSender, HelloHandler,

.

ListReqHandler ListReqSender .

ListResHandler, ListResSender.

Hello A ; 0; Hello B ;A; 1 B: A <- heard

Hello A ; B; 0;hash0; Hello B;A;1;hash1 A: B <-advanced ; B: A <- delayed (inconsistent if wrong hash)

LISTReq A;B; 0-1; ListRes B;A; 2; hash0; hash1 (hash0.A= hash0.B -> ok)

BlockReq A;B;0-1; BlockRes B;A ; blokc0;block1;

Hello A ; B; 1;hash1; Hello A ; B; 1;hash1; (A,B<- synchronized)

Blockchain structure :

Block2

Block1 :

BlockNumber ;

Hash Block father ;

Mining content (pow) + hashMining

Payload (signed)  + hashPayload

(Timestamp)

Global Hash :hash( BlockNumber,father,hashMining,hashPayload)

;

Synchronisation :

A num = 6, B num = 9.

A : HASH A;B;6. B HASHRES B;A;6;loremipsum;

Hash incorrect

Hash correct

Dowload next Fork

A: Dowload Block 6-9 from B Fork discovery