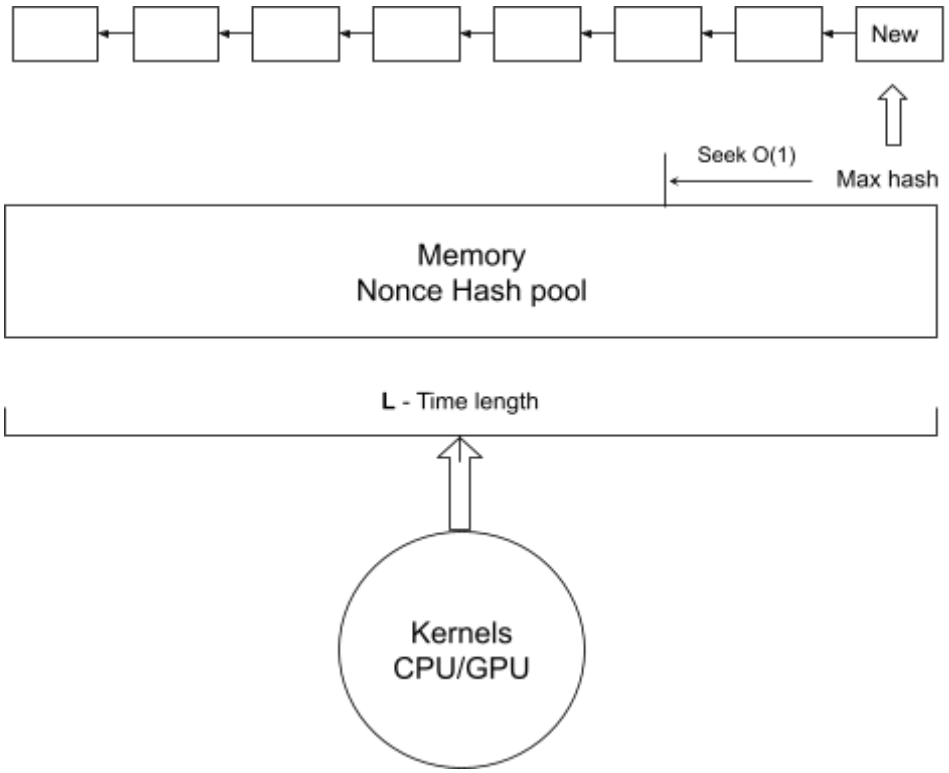
# Proof of Memory (PoM)

(ver. 0.2 from 30.05.2022  Mail: progr76@gmail.com)

Consensus is designed to significantly speed up the processing of transactions and blocks to the level of POS/POS consensus, but at the same time maintain the level of reliability and decentralization at the level of POW consensus. As is known, the disadvantage of POS is the dependence of the chain on the internal variables of the blockchain, which results in the cheapness of rewriting the transaction history. At the same time, in POW, the chain depends on the energy spent, which means changing the history is expensive. An additional bonus of the PoM consensus is the equalization of mining equipment based on GPU and ASIC, as well as unlimited scalable sharding.

To do this, we suggest using memory in addition to calculating hashes, but unlike other similar algorithms (for example, Ethash), in our algorithm memory does not limit, but improves the performance of the target equipment. This can be done due to the fact that when selecting a hash, not an integer nonce will be used, but a certain value that is time-consuming to calculate - for example, calculated using the SHA-3 algorithm, and repeated use of this value will be allowed for some time to select the best hash of the block. Thus, it will be more profitable to store these values in memory than to calculate them again.

A scheme for pumping memory with hashes and finding the best value for a new block:




1. CPU/GPU cores perform constant memory pumping with “fresh” hashes.
2. Periodically, the “best” hash for a new block of the chain is searched in a parallel stream.

3. If the core + memory configuration is not balanced, for example, there is not enough memory, then after filling the entire memory pool, computing cores can be stopped, which can lead to energy savings.

In this scheme, we use the time parameter L, which sets the validity of previously calculated hashes in the pool. In other words, it determines the degree of superiority of memory over computing. The hashes written to the buffer are quickly retrieved from memory each time the block hash is calculated, while pure computing equipment (ASIC) requires a new calculation.

## Leader Block search algorithm

The formula for calculating NonceHash for pumping pools in memory:

*NonceHash = sha3(Nonce, PubKey, LPowHash)*
*where:*
*Nonce - any integer number*
*PubKey - miner's public key*
*LPowHash - hash of a block with a number lagging behind the current one by no more than L*

To calculate the current hash, the miner uses a *NonceHash* from the pool and the formula looks like this:

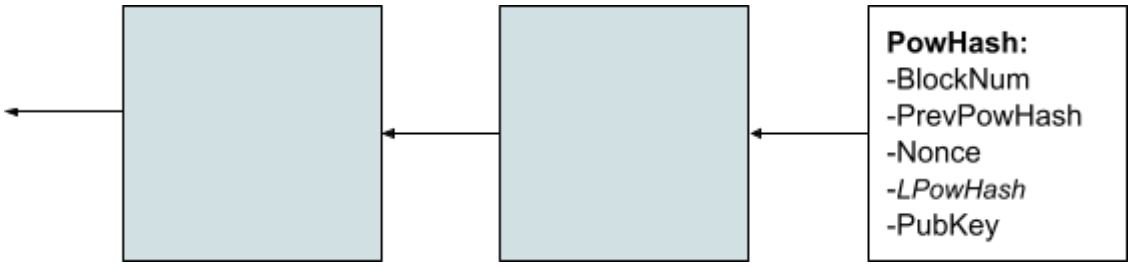
*PowHash= XOR(sha3(BlockNum,PrevPowHash), NonceHash)*
*where:*
*NonceHash - valid nonce hash from the pool*
*PrevPowHash - hash of the power from the previous block*
*BlockNum - current block numbers*

*<u>Features:</u>*
*The use of XOR suggests that for a high hash power (i.e., the number of first zeros), you need to find the closest value to each other, which allows you to search for a sorted pool of hashes in one memory access (because the time of the search algorithm is of the order of O(1)), and therefore reduce power consumption.*

Parameters are sent to the network: *BlockNum,PrevPowHash, Nonce*, *LPowHash, PubKey* which are proof of the work performed. Each node checks the validity *LPowHash* - it must be present in the chain, and also must not lag behind by more than L blocks and restores the hash power according to the formula:
*NonceHash = sha3(Nonce, LPowHash, PubKey)*
*PowHash = XOR(sha3(BlockNum,PrevPowHash), NonceHash)*

*Remark:*
*Instead of the LPowHash parameter, the block number containing it can be transmitted to the network.*

Block chain scheme (PoW chain):




## Block Data binding

The above algorithm determines the leader block, but does not specify the contents of the blocks. In order to bind the block data, the miner must sign the hash of the current block data with his private key. In this case, a signature block is added to the schema.

The formula for calculating the current hash of the data block:

*BlockHash = hash(BlockNum,PowHash,PrevBlockHash,DataHash)*
*where:*
*BlockNum - current block number*
*PowHash - hash of the power of the current block*
*PrevBlockHash - hash of the previous block*
*DataHash - hash of transactions of the current block =hash(TxArr)*

Signature calculation formula:

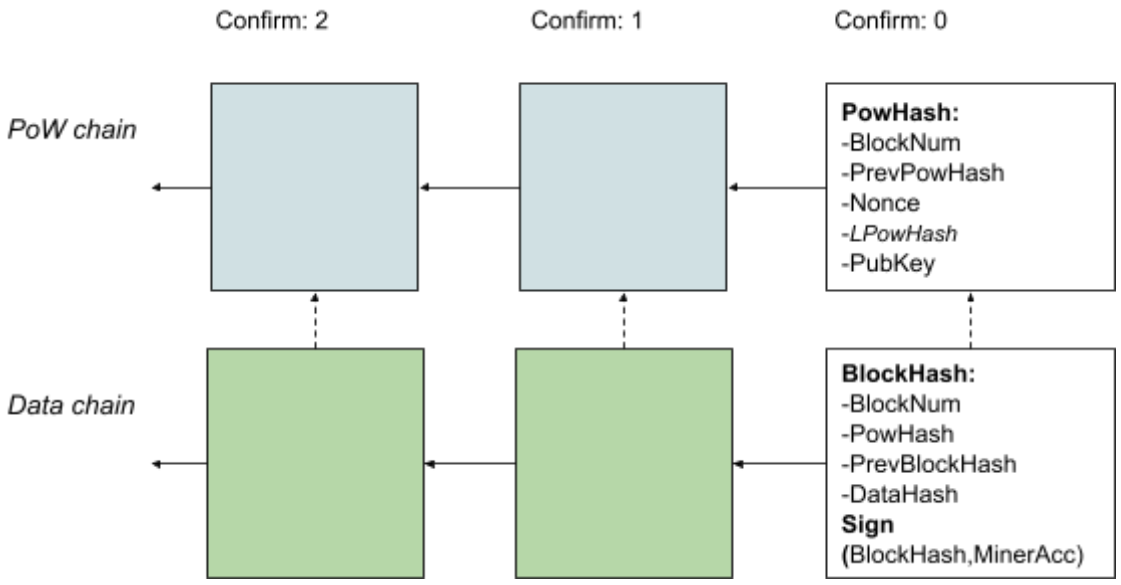
*Sign = sign(BlockHash , MinerAcc)*
*where:*
*BlockHash - block hash*
*MinerAcc - miner's account for block reward*

The same parameters as above are sent to the network: *BlockNum,PrevPowHash, Nonce, LPowHash, PubKey,* and also additionally: *PrevBlockHash,DataHash, MinerAcc,Sign.*
Where are the parameters: *MinerAcc,Sign* - they are optional (in this case, the miner does not receive awards, and the chain is confirmed by the following manners).

The procedure for performing checks:

1. The miner's public key is calculated from the parameter *Sign*
2. The correctness of the digital signature is determined
3. The hash power is calculated
4. A candidate for the block leader is being determined

The general scheme of the block chain:




Notes:

- Miners, by signing blocks, vote for a chain with data, where the weight of the vote is equal to the number of bits *PowHash*.
- Since in fact the communication of blocks with data is carried out in a separate chain, the block is considered confirmed if the next block with data is created and signed.

## Consensus Leader-chains

For consensus, the global network time is used, which is the same for all nodes. Since we know the start time of the network (the first block), the period of creation of one block and the current time (the last block), then all chains have the same length.
The chain that has the **maximum sum of the number of leading zeros** of hashes (*PowHash*) will be the leader.

## Sharding

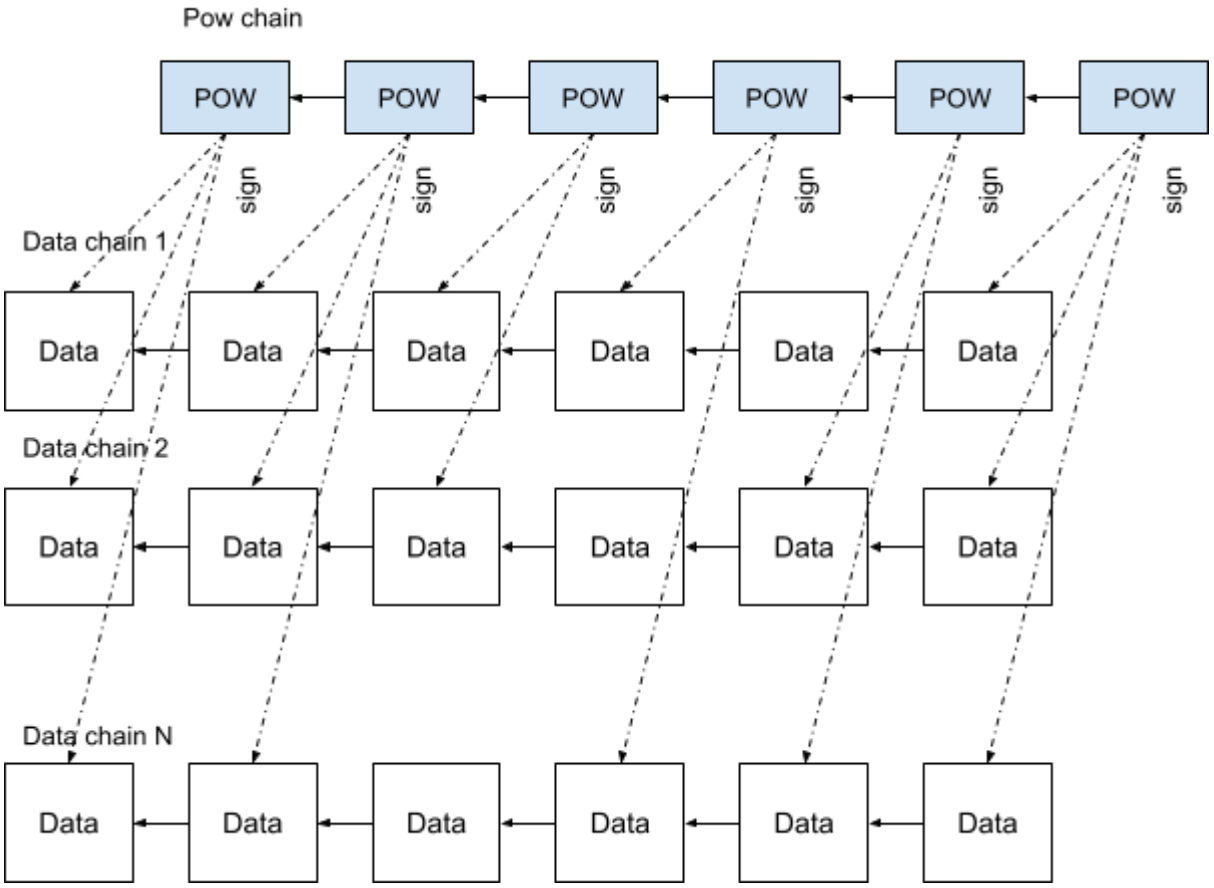
Here, sharding refers to the possibility of connecting blockchains based on PoM consensus with each other to create a single information space for the exchange of transactions.

### Common PowHash chain, different data chains

Each shard validates a common PowHash chain, which consists only of the header part, since it contains only fields: *BlockNum,PrevPowHash, Nonce*, *LPowHash, PubKey* which can be encoded in less than 80 bytes per block. The purpose of this chain is to determine the miners who have the right to sign blocks with data.

Each shard has its own data chain, which is signed by miners from the PowHash chain. The more miners there are in such a chain, the faster the confirmation of blocks with data takes place.

.

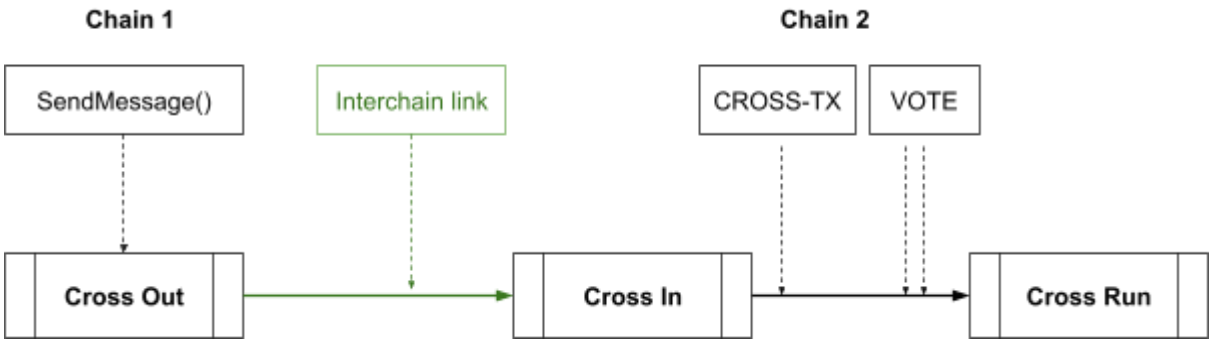



Data chains are essentially a separate blockchain with their own tokenomics and motivation for signing blocks by miners. Each new chain does not add electricity costs to calculate PoW, as a result, the specific energy efficiency increases and thus can reach the indicators of POS consensus.

## Cross-block whose new transactions (messages)

If we have a miner that validates several blockchains at the same time, then when forming block data, he can add special transactions to them that contain information about the presence of cross-messages in other chains. Such transactions by themselves do not participate in the exchange between nodes before the block is created, but can be obtained through the Interchain link - a network that connects the internal nodes of one miner.

Cross-message delivery scheme:

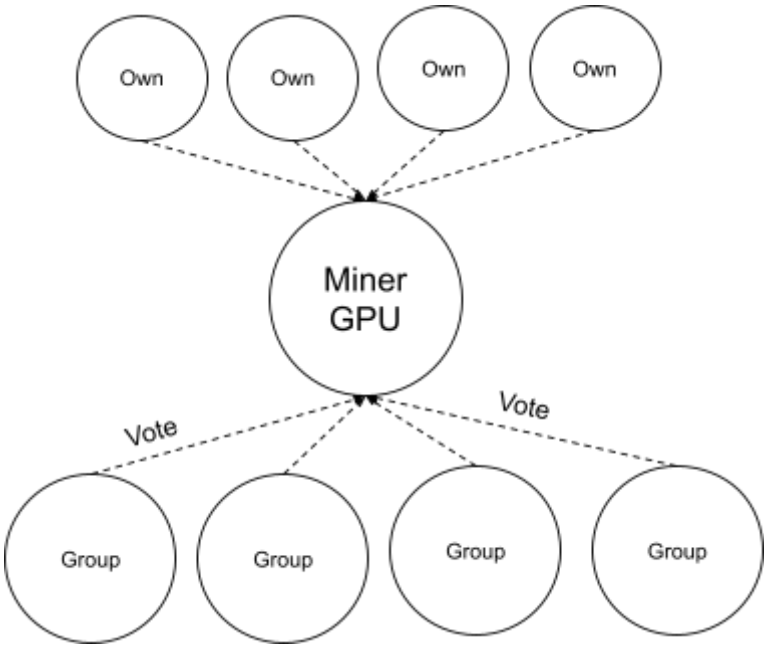



Miners perform voting for a particular composition of cross-messages through special voting transactions. The more votes a message contains, the greater the credibility. When a certain threshold is reached, the message is fixed in the receiver chain. In fact, miners act as oracles - witnesses of the presence of any information in the source chain.

Given that such a sharding scheme does not require a centralized chain, it means there is no “bottleneck” and sharding is **not limited in scalability**, while blockchains actually work in a single information space. Security is provided by all the general manners of the network

### Decentralization of miners

GPU software miners validate only the Pow chain, and data blocks are signed from information received from full nodes. Full nodes can be either of the same manner, or from different groups of nodes that he trusts, in order to increase security and decentralization, the choice of the correct data chain occurs through voting.



## Quick block creation

In this algorithm, you can set a very low block formation time, for example, 1 second or even less.
In the classical PoW scheme, the minimum time is limited by the time of calculating the block hash and the network speed for its distribution across all nodes. The problem is that in order to start forming and calculating a block, you need to have a current leader chain. If blocks are created very quickly, then network delays make the problem insurmountable. Therefore, the best time of all pow blockchains is 15 seconds, which is represented in the implementation of the Ethereum blockchain.

In the proposed PoM consensus, the block hash is searched almost instantly - O(1), which allows you to rebuild blocks (chain tails) to find the optimal chain (the sum of the capacities of all blocks is maximized).
In addition, the leader chain can be built in advance, which allows you to find the leader miner beforehand and send him transactions of the current block, so the speed of block creation and confirmation will be very high - no worse than in partially centralized POS/DPOS type consensuses.

## Security

### Changing the composition of transactions (data blocks)

Since the block data is bound by the miner arbitrarily (through his digital signature), it is important how many confirmations of this block were from other miners. If the next miner linked his blog to the previous one via the prevBlockHash parameter, then the data binding in the previous block can no longer be changed (without spending work), which means its transaction composition will remain unchanged.

Thus, the scheme requires one more confirmation, unlike the classic power scheme.

### Attack 51%

If an attacker has more than 50% of the network's power, then he can gain control over all the blocks being created. The attacker will add only his blocks to the chain, and due to the fact that his chain has more power on average, he will win on all blocks.
Transactions are linked to the block by the block miner by creating a digital signature that takes into account the reference to the data of the previous block. If the miner has created the last two blocks, then he can change the previous one and recalculate the next one. Thus, if a miner has created the last n blocks, then he can overwrite them all quickly enough, so such a chain is not reliable from retroactive changes.

Chain changes can be used in “double waste" attacks.

Protection against such attacks is based on the concept of the cost of the attack. As long as the attack is economically unprofitable, the valuables will be safe. Thus, the reliability of a transaction is determined by the number of confirming blocks, i.e. the time that has passed since it entered the blockchain. The **cost of an attack** is defined as the **price of renting** the equivalent capacity of the entire network during this time.