MEMCOIN2: A HYBRID PROOF OF WORK/PROOF OF STAKE CRYPTOCURRENCY

ADAM MACKENZIE[1]

Abstract. Cryptocurrencies are gaining traction as monetary instruments. A novel cryptocurrency with a hybrid proof of work and proof of stake system is proposed that affords eventual control of monetary supply to the userbase. Among the features included are a novel hash tree termed a polymorphic hash tree and extensive use of sequential memory-hard secure hash algorithms.

Introduction

The digital currency system Bitcoin (BTC) (Nakamoto, 2008) has, in five years time, become a vast global store of wealth. Similar cryptocurrencies, such as Litecoin (LTC), have utilized a sequential memory-hard secure hash algorithm (Percival, 2009) to make proof-of-work (PoW) mining of the chain less efficient for field-programmable gate array (FPGA) devices and application-specific integrated circuits (ASIC). Recently, a BTC derived cryptocurrency called Peer-toPeer Coin (PPC) (Nadel, 2012) was released that introduced a novel form of block generation known as proof-of-stake (PoS). The principles of both LTC and PPC are extended in Memcoin2 (MC2), a cryptocurrency system detailed within.

1. Polymorphic, sequential memory-hard secure hash trees for PoW

A novel approach to the PoW algorithm for MC2 is the use of a polymorphic, sequential memory-hard secure hash tree. BTC began with the use of SHA256 as the secure hash algorithm, followed by sCrypt (Salsa20/SHA2-256; N = 1024, p = 1, r = 1) for LTC. MC2 extends this approach by the use of an sCrypt utilizing **polymorphic hash tree** which has the following associated properties:

i. N, a component of the sCrypt algorithm that determines memory-hardness, are pseudorandomly selected for every n blocks of the chain within a given memory range Z. For MC2, Z = {64 KB, ... , 320 KB}, and we achieve our Z range by setting N = {512, ... , 2560} ($N_{range,Z}$) while leaving r = 1 and p = 1 as constants. The number of blocks per cycle, n, is 8 and constant.

ii. To enhance FPGA/ASIC resistance as well as fault tolerance, the hash tree incorporates the use of more than one secure hash function. Specifically, BLAKE512, SKEIN512, SHA3-512 (KECCAK512), and SHA2-512 are incorporated with both Salsa20 and Chacha20 stream ciphers. The value n is again used to determine the cycle size of the polymorphic hash chain, where each of the 8 possibilities of a sequential memory-hard hash function are incorporated into the hash chain once and only once; as before, in MC2 n = 8. These are also ordered in a pseudorandom fashion every cycle.

The question of pseudorandom selection of the components in both **i.** and **ii.** must now be addressed. At the end of the 8-block cycle, the selection of the values in the range specified in **i.** will be chosen from a series of 8 strong sequential memory-hard Chacha20/SHA3-256 (N = 262144, p = 1, r = 1) hashes of the last 256 bits from the last 8 block header hashes. Because of the difficulty in calculating this hash, the selection of block header hashes to allow the manipulation of N values should be also be difficult. The 256-bits returned from each of the 8 hashes is then used to select values in our N range such that the 8 subrange set of $N_{range,Z}$ containing $\{\{N_{min}, \ldots, N_{min}+1[(N_{max}-N_{min})/n]\}, \ldots, \{N_{min} + 7[(N_{max}-N_{min})/n], N_{min}+ 8[(N_{max}-N_{min})/n]\}$, termed N’, has each subrange

[1] tacotime at http://bitcointalk.org

selected once and only once. To clarify, N' in the case of MC2 = {{512, ... , 768}, ... , {2304, ... , 2560}}. We now assign an order to the the hashes based on the order of the magnitude of the 256-bit hash, where the smallest hash in value corresponds to the smallest subrange while the highest hash in value corresponds to the largest subrange (Figure 1).

Now that our range is selected, we use the 8-bit Pearson hash (P) of the original hash to determine the value within the range to set N to, such that $N = N_{\text{minimum in subrange}} + P$.

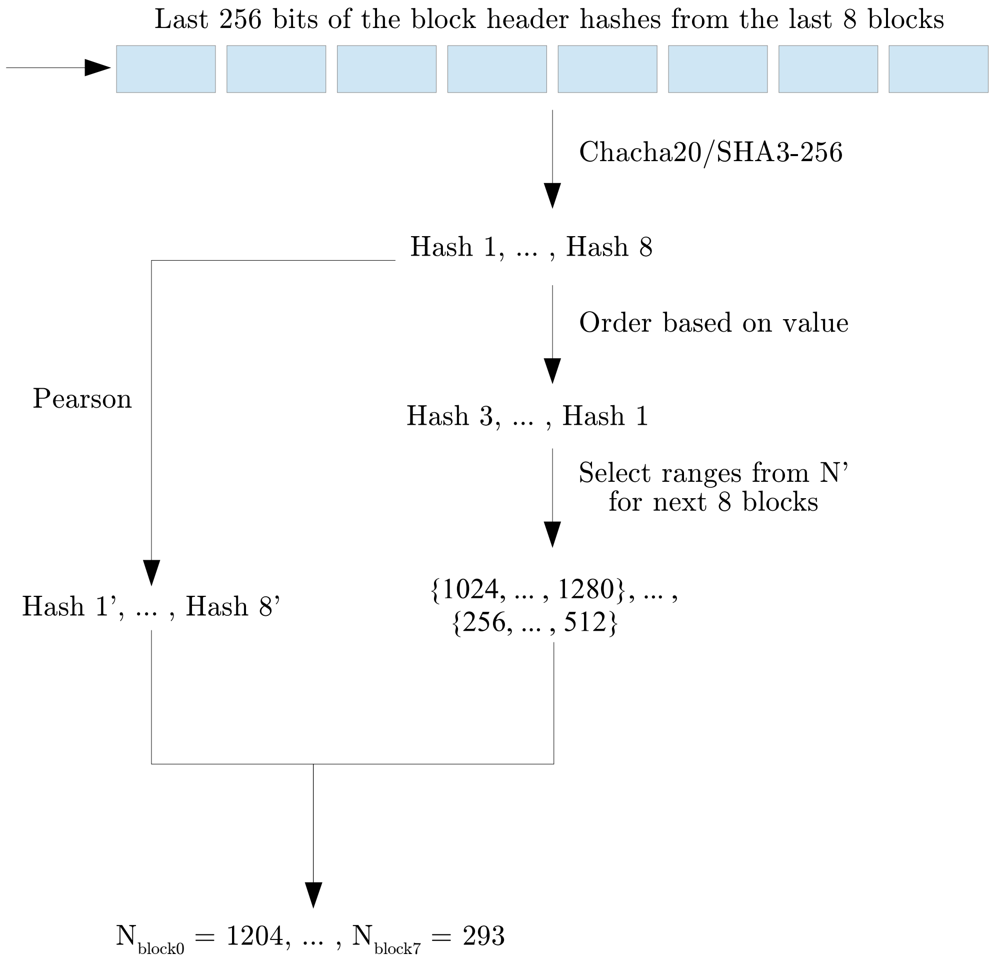

Figure 1. Schematic diagram demonstrating how the pseudorandom selection of N values for the next 8 blocks is achieved in MC2 using a difficult Chacha20/SHA3-256 hash followed by a Pearson hash. N' is the 8 subrange set of $N_{range,Z}$ containing $\{\{N_{min} , ... , N_{min}+1[(N_{max}-N_{min})/n]\}, ... , \{N_{min} + 7[(N_{max}-N_{min})/n], N_{min}+ 8[(N_{max}-N_{min})/n]\}$.

The next question of pseudorandom selection is for the order of the secure hash algorithm that we will use for each of the blocks in the 8 block cycle in MC2. For the first 360 days of the design (259,200 blocks as will be discussed later), the order will be determined from a pseudorandomly generated table that ships with the software. After this period, the order will be determined by the integer ordering of the Pearson hashes of the last 256 bits of the block header hashes of blocks {(current block − 259,200) ... (current block − 259,192)} such that

000-031 = Salsa20/BLAKE512
032-063 = Chacha20/BLAKE512

064-095 = Salsa20/SKEIN512
096-127 = Chacha20/SKEIN512
128-159 = Salsa20/SHA3-512
160-191 = Chacha20/SHA3-512
192-223 = Salsa20/SHA2-512
224-255 = Chacha20/SHA2-512

The reasoning for N being based on very recent blocks versus the secure hash algorithm being based on a table and eventually older blocks is to ensure the diversity of randomization as well as to make sure that the most complex element of the secure hash function, N, is difficult to predict. The overall intention of the polymorphic hash tree is to make optimization extremely difficult for any ASIC or FPGA device without requiring the size of a scratchpad for Salsa20/Chacha20 to be present at 320 KB (the maximum memory usage) and to also force them to incorporate logic circuits for two different stream ciphers and four different secure hash algorithms that are required to be performed in random order for each series of 8 blocks. On top of this, a CPU or GPU is likely required to calculate the Chacha20/SHA3-256 hash with N = 262144 required to determine the next quantity of memory usage. This further presents an obstacle to FPGA and ASIC devices but not to computer systems consisting of CPUs and GPUs.

An FPGA or ASIC implementation of a miner for this hashing algorithm should ideally include either an instruction cache (to contain instructions for the different hash functions each block) or all the hashing algorithms hard-coded as logic circuits. It should be noted that this algorithm will **not** solve the problem of the ability to mine with FPGAs or ASICs and only delays their adoption.

2. A novel PoS system that promotes value and prevents 51% attacks

The PoS system in MC2 is dissimilar to that within PPC and does not involve the solving of stake blocks. Instead, the stake algorithm is defined as follows:

1.) A new PoW block is mined and transmitted through the network containing at minimum the block header, coinbase transaction, and a 512-bit hash representing work at the current difficulty.

2.) An average of five stake signatories from a minimum of 29 days in the past will be selected based on the last 256 bits of the 512-bit block header hash and bits from the last 65,536 blocks (the lottery winner hash).

3.) The signatories each vote on the last block by signing transactions from their respective addresses. These transactions are fee free, but may only contain two transaction ledger hashes (previous block transaction ledger and lightweight whole chain ledger*), an address for stake reward, and a vote bit. The vote is based on the hashes of the transaction ledger that are generated by each of the signatory nodes during the time period in which the last block was generated (see below).
* = See section **6** of this paper.

4.) If the majority of signatories sign Yea on the last block (transaction ledger hashes match), the block is considered valid by the network and remains in the chain. However, if the last block is signed Nay by the majority of signatories, the block is considered to be malicious and all transactions are pruned from it. Pruned transactions must then be rebroadcast to be included in the block chain. The malicious block remains in the blockchain as it represents real work (contains a PoW hash of the block header that is valid) and increases the block height. As all transactions are not valid from this block, the PoW/PoS rewards in that block are also invalidated. However, PoS tickets used in an invalidated block are still considered spent and original sums of the PoS stake submission returned.

5.) Eventually, another PoW block is mined. This PoW block now contains the stake signatory transactions that were used to validate or invalidate the last block, and the stakeholders receive their reward for securing the PoW chain. Stakeholder tickets are considered used at this point, are removed from the pool of eligible stakeholder tickets, and may not be respent. Go to **1.)**.

There are several issues that must be addressed here for such a system to work. The first is how PoS holders are able to verify the transaction ledger of a PoW miner. To do so, we require that both the PoW miners and PoS signatories contain exactly the same transaction ledger. The easiest way to do so is to automatically sort the previous block transaction ledger by stake signatories first, stake submission transactions second, then by fees, then by address, excluding the PoW coinbase transaction (which is different for any given PoW miner). By sorting the hash tree of transactions in this way, we should get a congruent root hash between both the PoW miners and PoS signatories. PoS miners can keep a small record of the last several root hashes in the case that the root hash of the mined block does not contain all the latest transactions[2]. In this way, fees are also used to give priority to transactions, as transactions which do not contain a large enough fee may be bumped off the ledger if the block size exceeds that of its set limit.

The next issue is how to select the PoS signatories for each block. Signatory selection should be a random lottery to prevent monopolization of the network by smaller stakeholders. To select signatories pseudorandomly, the following method will be used:

1.) A stakeholder transmits a stake submission transaction to the network. This stake transaction includes all input addresses used for generating stake, and must contain a quantity of coins in the input equal to or greater than the **stakeholder difficulty**. The output of the transaction is the lowest value address of the input transactions. These are allowed to be submitted to the network fee free, as these transactions are used to secure the blockchain. Once the stakeholder submits these coins to the network as PoS, they are considered unspendable and removed from the balance of the stakeholder's wallet.
2.) After approximately 29 days (20,864 PoW blocks), a stakeholder transaction matures and is issued a **ticket**. The ticket is valid for 65,536 blocks, then expires except in the case that no lottery winner hash for the ticket has been drawn to allow for the redemption of it. In this case, an extension of 16,384 blocks is given. So long as no opportunity arises to spend, these extensions may be given indefinitely.
3.) Over the next approximately 91 days (65,536 PoW blocks), a stakeholder may use their ticket to vote on a previous block whenever the **lottery winner hash** indicates that they are eligible to do so (when the value of the ticket is equivalent to the value of the lottery winner hash). Because there are five intended signatories for each block, the amount awarded for claiming a stakeholder transaction is the stake reward divided by 5.

Definitions
Lottery winner hash: The lottery winner hash is calculated from the SHA3-16 hash (first 16-bits of a SHA3-256 hash) of a bit from 1,024 of the last 65,536 blocks' blockheader hashes.
Stakeholder Difficulty: The number of coins required to send a stake submission transaction through the network as a function of the fraction of total coins available. Like PoW difficulty, this is self-adjusting and is based

[2] Consistency issues may also arise; for example, if two transactions are sent at nearly the same time, Node A might first see Tx A and Node B might first see Tx B. Node A claims a block and hashes the transaction list with Tx A sorted into the list. Then Node B sees Tx A and makes a list sorted with both A and B in it. Node B will thus never generate the same hash as the one used by Node A to solve the block. To remedy this, stakeholder nodes will have to generate hashes for all possible incorporations for the last *n* transactions. Ideal *n* values will need to be experimentally determined and will change with the rate of transactions on the network.

on the linear weighted moving average of new stake submission transactions for the past 18 days, with a target of 5 new stake submission transactions per block.

Ticket: The ticket is used to redeem stake reward by acting as a stake signatory to some block. A ticket is calculated from the SHA3-16 hash of a bit from 512 of the next 20,864 blocks' blockheader hashes after the original stake transaction and the hash of the original stake transaction in the block ledger (the latter to prevent duplicate tickets from arising in the same block).

Testing will be required to demonstrate that the ticket and lottery winner hashes generated are both unique and pseudorandom. Given both these conditions are satisfied, the stakeholder will be able to use his ticket to sign a block within an average of approximately 74.5 coin days after the submission of their stake transaction. By 120 coin days (one third of a coin year) after submission of the stake transaction, the likelihood is very high that the stakeholder ticket will have had the option to sign a block.

At the introduction of any new block in the network, it will instantly be known how many stakeholders are available and eligible to sign this block. As soon as the block reaches a majority of Yea votes from stakeholders, it is accepted and locked into the block chain. If two blocks are solved at approximately the same time, the one with more Yea stake signatories is chosen as the valid chain upon the solvation of the next block. Because stake transactions may be near instantly seen on the network, miners will likely begin mining the chain which has more signatures visible and themselves propagate the chain with more stake transactions.

Because the valid chain is considered to now be the chain that is both the longest and has the most stake signatories, there are three things required to double spend: (1) the attacker should ideally have 51% control of the hashing power (2) the attacker should ideally have 51% of the stake of the coin and (3) the attacker must use their stake transactions in secret in addition to mining PoW blocks in secret. It be may be wise for clients to simply reject reorganizations of more than one or two blocks to prevent double spends, so long as there are at least a few stake signatories present on the most recently mined blocks. It should also be noted that a double spend can theoretically be performed with less than 51% of stake tickets being controlled by the attacker, but requires the attacker to mine with 51% of the network's hashing power for much longer periods of time given the random probability of the lottery system for stake.

Given the reduced likelihood of double spend that should be afforded by using stakeholders to sign new blocks into the network, it is likely that the transactions can be considered secure after 2-3 blocks have entered the network and been signed Yea by a majority of the total possible stakeholders. Blocks should be considered less secure if no stakeholders sign them before the next block or if the number of potential stakeholder signatures is less than 50% of the total possible stakeholders based on the number of tickets present for the lottery winner hash.

A malicious entity with 51% of stake can also perform a number of new attacks on the network. The foremost attack would be to continually invalidate blocks, stopping all network transaction traffic. This in itself is deleterious to the attacker, as it also invalidates their own stake rewards and spends their own stake tickets. Such an attack is thus unlikely. The attacker would also be able to select for which blocks are valid and which are invalid; in collusion with a large miner, this could allow the large miner to accumulate more coins which are fed back to the 51% stake holder to solidify their cartel. This is a much more likely attack. Defence against this attack vector will likely involve the network temporarily disabling PoS signing in favour of a pure PoW system or destroying all present PoS tickets and allowing tickets to only be generated using coins from more current blocks. Thus, a 51% stakeholder is now a more dangerous entity to the health of the blockchain than a 51% miner and the network must be vigilant to ensure that such a massive quantity of stake does not become concentrated.

There should be a means to discourage negligent stakeholder voting: for instance, voting Nay and submitting a random hash in conflict with the transaction ledger hash for the block or voting Yea and submitting the same transaction ledger as found in the block. To address this problem, stakeholders not voting in the majority will be penalized by having their stake reward destroyed upon the solvation of the block in which their stake signature transaction is found. This has the additional effect of making stake attackers less likely to submit stake transactions to the network for fear of losing their stake reward and ticket, and an absence of stake signatories for a long period of time may indicate that a malicious entity is acting upon the network. It should also be noted that even if the reward is destroyed, the stakeholder's original coins are still returned to them upon consumption of the ticket.

The last problem addresses the possibility of a blockchain fork arising from two blocks entering the network at exactly the same time and being signed by exactly the same number of signatories with exactly the same number of potential signatories. If two blocks have the same number of signatories but one has a lesser number of potential signatories (based on the lottery winner hash), we simply select this block. However, in the unlikely event that both the number of signatories are the same and the number of potential signatories are the same, the network simply selects the block with the larger amount of work present in the block header hash as being the correct block.

3. Miner reward algorithms and the rate of block and coin generation

The base rates for the MC2 network differ from those in previous cryptocurrencies in order to simplify the system: an MC2 year is 360 days (a coin year), difficulty adjustment periods are every 4.5 days, PoW and PoS block reward adjustment periods are every 9 days, and PoS average coin age requirements is 120 days (coin quarters). The network target of MC2 is 30 blocks per hour for PoW and 150 new stake submission transactions per hour for PoS.

The rate of progressively reduced block reward and constant block generation in BTC PoW block generation results in massive disinflation[3] as network time progresses. This is expected to promote the adoption and usage of BTC as a monetary instrument. By contrast, PPC uses a PoW block reward algorithm that can both inflate and disinflate depending on the network hash rate and makes the assumption that network hash rate will be consistently increasing exponentially. In addition, PoS blocks in PPC apply a 1% per annum interest to stake miners who obtain them. MC2 abandons PPC's PoS reward generation algorithm and employs a similar system of block reward seen in BTC and LTC for PoW blocks, with faster block reward adjustments. The average quantity of stake reward is calculated similarly. This provides a strong incentive for the mining of both PoW blocks and block confirmation by PoS stakeholders.

Reward adjustments are performed every 9 days resulting in an 8% decrease in value per 360 d MC2 annum (8% disinflation rate) starting with a base PoW reward of 25 coins per block and a base PoS reward of 12.5 coins per block by the equation

$$(1)\quad R_{\text{current}}=R_{\text{initial}}-R_{\text{initial}}\left(1-\left(1+\frac{-0.08}{40}\right)^{\left\lfloor\frac{\text{block height}}{6480}\right\rfloor}\right) .$$

A lower starting reward for stake is given to ensure that the incentive for PoW miners is always greater than that for PoS miners, as the former perform more computational work. Given a target rate of 30 blocks h^{-1}, the chain should approach a daily amortized inflation rate of approximate 1% (Figure 2; see below) as it reaches coin year 27 or block 6,998,400. At this point, the interest rate stemming for both PoW and PoS rewards become locked in for one coin year and is then controlled by the userbase rather than being hardcoded into the chain itself. Yea or nay votes are given with each block obtained in the block header; at the end of each coin year from block 6,998,400 onwards,

[3] Please note that inflation in this article refers to the increase in supply of a given cryptocurrency, while disinflation refers to the decrease in supply inflation of a cryptocurrency.

the sum of all votes for both PoW and PoS are tallied as yea (1) or nay (-1), summed, and divided by the total number of votes (blocks for this coin year) to yield a elected fraction *v*. This fraction is then the difference in inflation applied to each work system for the following coin year, with a maximum difference of 1% total.

A requirement for the PoS inflation percentage is that it may not be democratically elect a PoS reward to a value of more than 50% of the PoW reward by inflation, to prevent older PoS signatories from overtaking the chain and affording control of the network to a group of hoarders who perform virtually no work (a problem observed with current global financial systems). A requirement for both the PoW and PoS inflation rates is that they do not go below 0%, as this removes incentive to mine or sign PoW blocks.

We can represent the overall inflation of chains mathematically as an amortized inflation rate per year f_i for a given time period of days d_T with a daily coin generation rate b_r from start date in days of d_i as

$$(2) \quad f_i = \frac{1}{d_T} \sum_{n=d_i}^{d_i+d_T} \frac{b_r}{nb_r} \quad .$$

By this metric, we can observe that the rate of disinflation for MC2 is less than that for BTC and LTC beyond 5 years post-conception. Dissimilar to PPC's constant PoS inflation rate, the PoS rate of inflation in MC2 is precisely the same as that of the PoW until reaching coin year 27, at which time the rate is frozen for one coin year and then handed over to the democratic control of the miners.

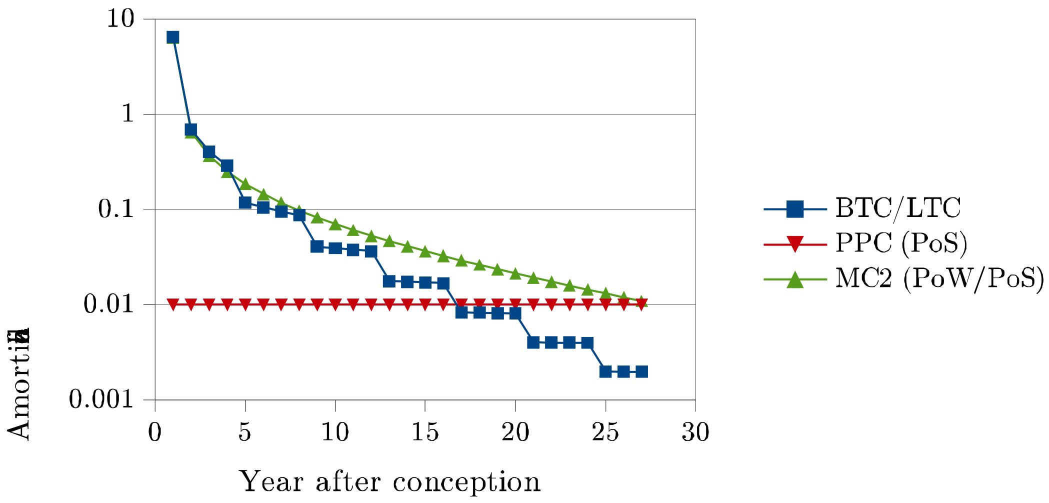

Figure 2. Amortized rates of inflation for various cryptocurrencies for a given year (365 days). Note that the amortized rate of inflation represents a yearly inflation rate calculated from daily rates of inflation as described by equation (2).

The difficulty for MC2's PoW chain is calculated from the linear weighted moving average of block times the past 18 days. The stakeholder difficulty is likewise calculated from the linear weighted moving average of new stake submission transactions per block for the past 18 days.

4. Modifications to the block header and transaction data

The block header will contain an extra eight bits that specify the following:

Bits 0-2: Which of the 8 sCrypt algorithms is used for a PoW block

Bit 3:	Voting yea or nay for an increase in PoW inflation
Bit 4:	Voting yea or nay for an increase in PoS inflation
Bit 5:	Unused (Vote bit; other possible issue in future)
Bit 6:	Unused (Vote bit; other possible issue in future)
Bit 7:	Unused (Vote bit; other possible issue in future)
Bits 8-263:	Lightweight whole chain ledger (see section **6.**)

The first three bits are necessary for clients who are downloading and verifying the blockchain to be able to easily do so. Without these being specified, the client will have to do a number of hard Chacha20/SHA3-256 (N = 262144, p = 1, r = 1) hashes equivalent to the block height to determine which N value to use for the blocks in the chain.

Vote bits 3 and 4 only affect the block during democratic voting cycles for the rate of inflation (see section **3.**).

Transactions will largely stay the same as in BTC; coin age will be calculated from the the timestamp of the block in which it appears. Clients will be able to verify the legitimacy of the stake block based on the network time of the blocks in which the inputs first arose, preventing a need to individually timestamp transactions.

Stake submission transactions are similar to coinbase transactions except they are required to have inputs (up to 256). These inputs must follow the rules specified in section **2**. The exact reward claimed in the coinbase will be easy to determine and verify based on the reward algorithm from section **3**.

Stake signatory transactions contain 512 bits for the previous block transaction ledger and the lightweight whole chain ledger, 6 bits for voting (as above plus 1 for voting Yea/Nay on the previous block), and are signed by the address of the lowest value input of the original stake submission transaction.

5. Coloured coins

The introduction of arbitrarily generated "coloured" coins mapping to an originator address can serve at least a couple useful purposes. The first is so that any organization can generate their own coins easily and trade them with clients over the network. Once example would be an organization that wishes to reward users for solving a problem of biological significance such as finding the minimum energy structure for a given protein. The organization could use their own centralized network to determine the reward for their client, then transmit the equivalent number of coloured coins to them over the network. Note that the organization would still be required to pay the fees associated with the network in network coins.

The second useful purpose is for decentralized exchange using the network as an escrow. This is performed using a unique transaction script called a **conditional transaction**. Exchange would use two conditional transactions: one for a coloured coin, and one for another coloured coin or for network coins. The transactions would send coins to their corresponding output addresses if and only if the corresponding transaction to their input is included in the same block.

Coloured coin genesis transactions would include the following information:

1) A script element indicating a coloured coin genesis transaction.
2) No inputs.
3) Only a single output to themselves.
4) The number of coloured coins to generate (to some predetermined limit).

After the genesis transaction, the coloured coins could be spent as normal coins; however, the coloured coins would have to be transmitted throughout the network with a special field known as the originator address that specifies the address of the genesis transaction output and a script flag indicating that they are coloured. Tracking coloured coins in the network would be optional for the client, who is able to specify which coloured coins to watch by address. Usage of coloured coins requires the full weight client, as they are not included in the lightweight whole chain ledger.

6. Lightweight clients

One problem with Bitcoin is the massively growing blockchain size. The solution herein involves the generation of lightweight whole chain ledger. The lightweight whole chain ledger contains the following information:

1) The SHA3-256 hash of this ledger and the block height at which it is found.
2) All addresses containing more than 0.0001 coins, the quantity of coins in these addresses (minus coins reserved by stake submission transactions), and the block heights in which the number of coins in the address last changed. The addresses are sorted by value of the address.
3) Block header hashes used as PoW for every block.
4) All currently submitted and unredeemed stakeholder transactions, their tickets (if applicable), and whether or not they have had the opportunity for redemption (their ticket was selected but they chose not to redeem it, which determines long term eligibility).

The list of addresses and associated block height for last transaction is stored as a sorted B+ tree on disk for both ease of access and simplicity in the addition of new addresses. This list is hashed by both PoW miners and PoS stakeholders, and are must be congruent among both. As with the block transaction ledger verified by PoS stakeholders, stakeholders must recalculate and store the hash of the tree after the addition of any individual transaction and store a number of them temporarily in case they are not included at the time the PoW block is solved[4].

To enhance security, the lightweight client should also hold the last 256 blocks in memory as well as the transaction lists the the previous block (which the network almost completely reverts to in the event the block is voted against by stakeholders and which may be required in the event that some inconsistency arises). Upon the addition of a new block, the last block is in the 256 block high list pruned. In the event of a large reorganization (unlikely), the list will need to be reacquired from one of the full weight clients, which can recalculate and redistribute them.

References

Nadal, Scott. 2012. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. Self-published.

Nakamoto, Satoshi. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. Self-published.

Percival, Colin. 2009. Stronger Key Derivation via Sequential Memory-Hard Functions. Self-published.

[4] Footnote 2 also applies here, but addition and removal of nodes from a B+ tree is $O(\log_b n)$.