Proof of Staked Work: A Simple PoW/PoS Hybrid Consensus

Qi Zhou
QuarkChain Foundation LTD.
Singapore, Singapore
qizhou@quarkchain.org

***Abstract*—Proof of work (PoW) and proof of stake (PoS) are two well-adopted consensuses in permission-less blockchains. With high hash power of a PoW network, PoW is secure at the cost of high energy consumption. The high energy consumption could be significantly lowered by PoS, which suffers from various attacks such as stake-at-nothing and fake stake. To harvest the benefits of PoW and PoS, we propose a hybrid PoW/PoS consensus algorithm - proof of staked work (PoSW). To produce a block, PoSW requires the block producer to lock some stake before mining the block. In addition, the number of required stake increases as the percentage of the hash power of the block producer increases. As a result, the cost of attacker (51% attack) could be potentially significantly increased. We propose a simple PoSW implementation based on account-based blockchain ledger model, and study its performance via Monte-Carlo simulations.**

***Index Terms*—cryptocurrency, consensus, proof of work, proof of stake**

I. Introduction

Currently, the major consensuses in public blockchains are proof of work (PoW) [1], [2] and proof of stake (PoS) [3], [4] and its variants (e.g., delegated PoS [5]). Since its debut from Satoshis famous white paper [1], PoW has demonstrated its strong security; however, it is slow and energy inefficient. PoS is more energy efficient; however, it suffers from several security concerns such as nothing at stake, staking grind, long-range attack, and fake stake [6]–[8]. To harvest the benefits of both PoW and PoS, hybrid PoW/PoS [9], [10] such as Dash [11] and Decred [12] are proposed. Dash incentivizes users to set up a masternode by staking 1000 DASH, and allows almost instant payment via InstantSend. However, Dash still suffers from the famous 51% double-spending attack. In Decred, every PoW-mined block must be signed by multiple stakers before appending to the chain, and thus Decred achieves better security (in terms of 51% attack) when the stakers are sufficiently decentralized.

In this paper, we propose a simple hybrid PoW/PoS - proof of staked work (PoSW). The basic idea of PoSW is that, if a miner wants to contribute its all hash power to the network (suppose p percent of all hash power of the network), the miner must stake the number of tokens that is proportional to p. This means that to perform a double-spending attack, the attacker has to stake some amount of tokens besides 51% of the hash power of the network. Since the staked tokens are locked and unstaking takes a considerable duration of time (e.g., a few days), if the attack is detected before the tokens are unlocked, the staked tokens will be slashed. In addition, PoSW may encourage more decentralization since a mining pool will be more costly to run. This is because the owner of the pool must acquire sufficient stake before maximizing the efficiency of all hash power the pool collected from its users.

The rest of the paper is organized as follows. Section II introduces the consensus model of PoSW. Section III presents an implementation of PoSW. Section IV gives some implementation results based on Monte-Carlo simulation. Section V concludes the paper.

II. Consensus Model

Suppose the hash power of a miner i in the network at a specific time is h_i (in terms of hash per second), the PoSW is formulated as the following problem:

$$\begin{aligned} \text{Maximize} \quad & H' \\ \text{subject to} \quad & H' = \sum_i h'_i \\ & h'_i \le h_i, \forall i \\ & h'_i \le f(s_i) H', \forall i \end{aligned} \tag{1}$$

where h_i is the effective hash power of the ith miner, H is the total effective hash power of the network, s_i is the stake of the miner, and $f(s_i)$ is the maximum percentage of the hash power miner i could contribute to the network, namely, **allowance**. The allowance function $f(x)$ should have the following two properties:

- $f(x)$ is a non-decreasing function, which means the more tokens the miner stake, the higher percentage of hash power the miner could contribute; and
- $f(x)$ is a super-additive function, i.e., $f(x) + f(y) \le f(x+y)$. This makes sure that a miner cannot get more allowance by splitting its stake.

A simple example of $f(s_i)$ is $\min(\alpha s_i/S, 100\%)$, where S is the total number of circulated tokens, and α is a system-wide constant. To simplify the model, we will use this example equation in the rest of the paper.

Problem (1) is a linear programming (LP) problem, which can be efficiently solved by LP solvers in polynomial time.

A. Example

To better understand the model, let us put some example numbers into the model and solve the problem: suppose there

are four miners with each of them having equal hash power of the network, e.g., 25 H/s, and the percentage stake of each miner is $[5\%, 10\%, 25\%, 60\%]$ and $\alpha = 2$, then the PoSW model in (1) becomes:

$$\begin{aligned} \text{Maximize} \quad & H' \\ \text{subject to} \quad & H' = \sum_i h'_i \\ & h'_i \leq 25, \forall i \\ & h'_1 \leq 10\% H' \\ & h'_2 \leq 20\% H' \\ & h'_3 \leq 50\% H' \\ & h'_4 \leq 100\% H' \end{aligned} \tag{2}$$

By solving the problem in (2), the optimal solution is $h'_1 = 7.14286, h'_2 = 14.2857, h'_3 = 25, h'_4 = 25$ and $H' = 71.4286$. The percentage of effective hash power contributed to the network by each miner is: $[10\%, 20\%, 35\%, 35\%]$.

B. Discussions

From the result, we observe that:

- Even all of the miners have the same hash power, their effective hash power is bounded by their allowances (and thus stakes). This encourages a miner with large hash power and an insufficient stake to acquire more stake to maximize its effective hash power.
- The total effective hash power could be lower than the total hash power. This means a miner with sufficient stake may be easier to mine if other miners do not have sufficient stake.
- A double-spending attack requires 51% of the effective hash power and $1/\alpha$ percent of the circulated tokens in order to create a fork mined by the attacker solely.
- In addition, the stake that is used to mine will be locked for a specific period of time (e.g., 3 days). Therefore, if a double-spending attack is detected, the attacker can be penalized by slashing their locked stakes.
- By adjusting α, we could tune the system to be more in the favor of hash power or stake.
- A pre-mining step can be performed in the genesis block so that the pre-determined miners could have sufficient stakes to mine when the network just starts.
- It is possible that given a list of hash power and stake, and α, the problem does not have a feasible solution. For example, all miners do not have sufficient stakes, while the stakers do not want to mine the network. This is similar to the availability problem of validator in a pure PoS system. However, in this case, the stakers will be incentivized to mine the network because of the low hash power of the network. To further eliminate such case, we may still allow a miner to produce a block but the block must reach a much higher difficulty than other miners with sufficient stakes (e.g., β times higher difficulty than other miners with sufficient stake, where $\beta > 1$).
- A mining pool in PoSW needs significant more cost compared to PoW. To efficiently mine blocks, depending on the percentage of the effective hash power of the pool, the pool has to acquire the corresponding stakes in the network. In contrast, the running cost of a decent-size mining pool in PoW is almost negligible compared to the cost of the hash power the pool collected.

III. Implementation of Proof of Staked Work

In this section, we illustrate a simple and approximate implementation of PoSW. We consider an account-based blockchain ledger similar to Ethereum. First of all, we estimate the percentage of the effective hash power of a miner i over the network $p_i^{(j)} = \frac{h_i'^{(j)}}{H'^{(j)}}$ at the block with height j by dividing the number of blocks produced by the miner in a recent window with window size w:

$$\tilde{p}_i^{(j)} = \frac{\sum_{k=0}^{w-1} \text{iif}(c_{j-k} == c_j, 1, 0)}{w} \tag{3}$$

where w is the window size, c_k is the coinbase address of the kth block, c_j is the coinbase address of miner i, $\text{iff}(x, y, z)$ returns y if x is true else returns z, and the summation calculates the number of blocks mined by miner i in the window. Note that the estimator in Eq. (3) assumes that the hash powers of all miners are coherence over the window.

To enforce PoSW, the implementation requires that for a block with height j, a valid block produced by a miner must satisfy

$$\tilde{p}_i^{(j)} \leq f(s_i^{(j)}) \tag{4}$$

where $s_i^{(j)}$ is the balance of miner i's address at block j. A couple of comments are of interest:

- Assuming the total effective hash power is constant over the window, we could improve the estimation of the percentage of effective hash power of miner i $p_i^{(j)}$ by increasing the window size. We will study the effect of the window size in the simulation.
- To avoid the miner circumventing the allowance constraint by transferring the stake to another address during the window (and thus mine using the new address), the stake (i.e., balance) of the coinbase address will be locked until at least the window is expired.
- To eliminate the case that all miners cannot satisfy Eq. (4), we could introduce difficulty penalty for those miners with insufficient stake. Suppose the target difficulty of block j is d_j. A block producer could produce a block if the block reaches the following effective difficulty:

$$d'_j = \begin{cases} d_j, & \text{if Eq. (4) is satisfied} \\ \beta d_j, & \text{otherwise.} \end{cases} \tag{5}$$

If β goes to ∞, then PoSW with difficulty penalty degenerates to normal PoSW. Note that for PoS with difficulty penalty, the optimal mining strategy of a miner i is to use two addresses $a_i^{(0)}$ and $a_i^{(1)}$, where $a_i^{(0)}$ is an

address with stake, and $a_i^{(1)}$ is an address without stake (or minimal stake). Then, the miner chooses the following coinbase address when mining block j:

$$c_j = \begin{cases} a_i^{(0)}, & \text{if Eq. (4) is satisfied} \\ a_i^{(1)}, & \text{otherwise.} \end{cases} \quad (6)$$

IV. Simulations

In this section, we study the performance of the proposed PoSW implementation via Monte-Carlo simulations. The simulator accepts the following parameters as input:

1) hash power (in terms of hash per second) of each miner h_i's;
2) window size w;
3) maximum blocks could be produced by a miner in any window, derived from their stake s_i's, circulated tokens S, and α;
4) difficulty penalty β if the number of blocks produced by a miner exceeds its allowance; and
5) number of blocks simulated.

To produce each block, the simulator will exclude the miners that do not satisfy Eq. (4) by checking the number of blocks produced by each miner in the recent window and comparing its allowance. Then the simulator randomly selects the block producer in the rest miners weighted by their hash power. Unless otherwise specified, the default number of blocks to simulate is 100K. To simplify the simulation, block propagation latency is ignored. We further assume that the block reward during the simulation is small enough that the reward does not change the allowance of all miners.

A. Simulation 1: PoSW with Different Allowances and Equal Hash Power

We simulate the example in Eq. (2) with equal hash power, stake $[5\%, 10\%, 25\%, 60\%]$, $\alpha = 2$, and the window size being 128. The allowance list is $[10\%, 20\%, 50\%, 100\%]$. A simulation result of the percentage of actual blocks produced by each miner is: $[9.07\%, 18.62\%, 36.23\%, 36.08\%]$, where we observe that all miners have a close result to the expected result $[10\%, 20\%, 35\%, 35\%]$, although the percentages of miner 1 and miner 2 are slightly smaller than expected. This is because the window size 128 is small, and as a result, the granularity of the estimate of effective hash power is too coarse. To further study the effect of window size, Table I lists the results of different window sizes:

Window size	64	128	256	512	Theory
Miner 1	8.79%	9.07%	9.59%	9.87%	10%
Miner 2	17.29%	18.62%	19.40%	19.64%	20%
Miner 3	36.74%	36.23%	35.57%	35.16%	35%
Miner 4	37.18%	36.08%	35.44%	35.32%	35%

TABLE I
Distribution of Block Production with Various window sizes

From Table I, it is clear that by increasing the window size, the percentage of blocks produced by each miner will be closer to the expected one.

If a miner exceeds its allowance and we allow the miner to continue mining with higher difficulty (by a factor of β), Table II summarizes the percentage of blocks produced with various β's, windows size = 256, and optimal mining strategy in Eq. (6).

β	2	5	10	20	∞	Theory
Miner 1	18.97%	14.00%	11.95%	10.81%	9.59%	10%
Miner 2	23.93%	21.92%	20.85%	20.18%	19.40%	20%
Miner 3	28.77%	32.05%	33.58%	34.36%	35.57%	35%
Miner 4	28.33%	32.03%	33.62%	34.65%	35.44%	35%

TABLE II
Distribution of Block Production with Various β's

From Table II, we observe that when $\beta = 2$, the miner with small allowance will produce much more blocks, while when $\beta = 10$, the percentage of the blocks produced by miners is close to that of $\beta = \infty$.

B. Simulation 2: PoSW with Different Hash Powers and Equal Allowances

In this subsection, we simulate a network with hash power $[100, 200, 400, 800, 1600]$, equal stake (20%), $\alpha = 2$, and the window size being 256. The allowance list is $[40\%, 40\%, 40\%, 40\%, 40\%]$. A simulation result of the percentage of actual blocks produced by each miner is $[4.01\%, 8.09\%, 16.22\%, 32.51\%, 39.18\%]$, where the expected result is $[4\%, 8\%, 16\%, 32\%, 40\%]$. Note that, the percentage of the miner with the largest hash power is slightly smaller than the expected value.

Table III further summarizes the percentages of blocks produced with various β's and windows size = 256. From Table III, when $\beta = 10$, the percentage of the blocks produced by miners is very close to that of $\beta = \infty$.

β	2	5	10	20	∞	Theory
Miner 1	3.53%	3.67%	3.87%	3.98%	4.01%	4%
Miner 2	6.98%	7.46%	7.97%	8.00%	8.09%	8%
Miner 3	13.90%	14.84%	15.55%	15.97%	16.22%	16%
Miner 4	28.25%	29.88%	31.13%	31.67%	32.51%	32%
Miner 5	47.34%	44.15%	41.48%	40.37%	39.18%	40%

TABLE III
Distribution of Block Production with Different Hash Powers and Equal Allowances, and Various β's

V. Conclusion

In this paper, we introduce proof of staked work (PoSW), where to maximize the effective hash power that a miner could contribute to the network, a miner has to acquire some stakes proportional to the percentage of the effective hash power of the miner in the network. A linear programming (LP) model is formulated to describe PoSW, and a simple and approximate implementation is proposed. A simulator is created to verify the performance of the proposed implementation and to show that the implementation could yield a very close result to the LP model.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2009.

[2] V. Buterin, "A next-generation smart contract and decentralized application platform," 2014.

[3] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake."

[4] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A provably secure proof-of-stake blockchain protocol," in *Annual International Cryptology Conference*. Springer, 2017, pp. 357–388.

[5] D. Larimer, "Delegated proof-of-stake white paper," 2014.

[6] A. Poelstra, "On stake and consensus," 2015.

[7] A. G. Iddo Bentov and A. Mizrahi, "Cryptocurrencies without proof of work," 2015.

[8] D. S. Lab, "fake stake attacks on chain-based proof-of-stake cryptocurrencies," https://medium.com/@dsl_uiuc/fake-stake-attacks-on-chain-based-proof-of-stake-cryptocurrencies-b8b05723f806, accessed 22 February, 2019.

[9] R. P. dos Santos and M. Swan, "Pow, pos, & hybrid protocols: A matter of complexity?" *CoRR*, vol. abs/1805.08674, 2018. [Online]. Available: http://arxiv.org/abs/1805.08674

[10] Z. Cheng, G. Wu, H. Wu, M. Zhao, L. Zhao, and Q. Cai, "Deterministic proof of work," *CoRR*, vol. abs/1808.04142, 2018. [Online]. Available: http://arxiv.org/abs/1808.04142

[11] E. Duffield and D. Diaz, "Dash: A privacycentric cryptocurrency," 2015.

[12] C. Jepson, "Decred technical brief," 2015.