Single-tiered hybrid PoW consensus protocol to encourage decentralization in bitcoin

Gyu Chol Kim*
Faculty of Information Science and Technology, Kim Chaek University of Technology, Democratic People's Republic of Korea
Corresponding author: kgc841110@star-co.net.kp

ABSTRACT We propose a single-tiered hybrid Proof-of-Work consensus protocol to encourage decentralization in bitcoin. Our new mechanism comprises coupled puzzles of which properties differ from each other; the one is the extant outsourceable bitcoin puzzle while the other is non-outsourceable. Our new protocol enables miners to solve either puzzle as they want; therefore, blocks can be generated by either puzzle. Our hybrid consensus can be successfully implemented in bitcoin, because it is backward-compatible with existing bitcoin mining equipment(more precisely, existing bitcoin mining ASICs).

Keyword Bitcoin, Proof-of-Work, consensus, decentralization

1. INTRODUCTION

Blockchain is a distributed ledger that is shared and maintained by all participants in the network based on a consensus protocol. The most widely used consensus mechanism is Proof-of-Work (PoW)[1], which has been deployed in public blockchain networks like Bitcoin[2] and Ethereum[3].

In PoW, block generation requires solving a cryptographic math puzzle whose solution is easy to verify but extremely hard to solve. The participants in the blockchain network exhaust their computing resources to solve the puzzle. Here, the generating of blocks is called mining and the participants are called miners.

Recently, the miners for PoW-based cryptocurrencies are centralized in the mining pool and so, mining becomes a competition between mining pools. However, mining pools undermine the decentralization and security of Blockchain. To discourage the pooled mining, a number of techniques such as 2-Phase Proof-of-Work(2P-PoW)[4], Sign to Mine[5], Non-outsourceable Scratch-Off Puzzles[6], PieceWork[7], Autolykos[8,9] and SmartPool [10] have been proposed. SmartPool implements a decentralized mining pool through an Ethereum smart contract whereas the others discourage mining pools through the non-outsourceable PoW puzzles, in which the entity solving the puzzle can steal rewards from the pool manager.

Another threat to decentralization came from the fact that ASIC-equipped miners are able to find PoW solutions much faster and more efficiently than miners equipped with the commodity hardware such as CPU, GPU and so on. To reduce the disparity between the ASICs and regular hardware, studies on memory-bound computations have been carried out[8,11,12]. The most interesting practical examples are two asymmetric memory-hard PoW schemes in which memory required to verify a solution is significantly less than to find it [11,12].

Meanwhile, in order to be used in bitcoin practically, any solution to the centralized mining problem must preserve the existing blockchain; preserve large investments many miners have made and are planning to make in their equipment; provide a seamless transition from the existing system to the new one, providing adjustable knobs that can be fine-tuned for a desired trade off that fits the community's needs[4]. However, none of the proposals of [4-10] satisfies all the requirements above. In other words, preceding techniques require some changes in design of the cryptocurrency and so, they are not compatible with the current bitcoin system.

The aim of the present work is to propose the practical solution to the centralized mining problem that satisfies every requirement above. We first propose a non-outsourceable PoW puzzle that is simple and based on hashed rate of individual miners. Our scheme works by adding PubKey(public key) and HeaderSig(signature) fields to the original bitcoin block header. In this scheme, miners should repeatedly produce a signature(HeaderSig) for the contents of the block header (everything except the public key and signature field), which is verified by public key(PubKey), and double hash the entire block header(including PubKey and HeaderSig fields) until the resulting hash is less than the difficulty. In this case, block reward of the coinbase transaction must be sent to only one Pay-to-Public-key-Hash(P2PKH) address which is produced from PubKey. Though the miners can generate many signatures without changing nonce field (ECDSA is randomized signature scheme and so, it is possible to generate many signatures for a single message and key pair), we leave the nonce and extra nonce in block header and coinbase transaction, respectively. The reason is that in order to generate a large number of ECDSA signatures for the double hash test, it is efficient for the miner to vary only the message through the nonce and extra nonce fields (See Section2 for more details.).

Second, we propose a single-tiered hybrid PoW consensus protocol by mixing our non-outsourceable PoW with the original bitcoin PoW. In our hybrid scheme, only one of the original bitcoin PoW and proposed non-outsourceable PoW is used to mine a single block.

Note. The hybrid consensus protocols such as Snow White[13], Proof of activity[14], Casper[15], PeerConsensus[16], Hybrid consensus protocol[17,18], Tendermint[18], Proof of authority[19], delegated proof of stake[20] and Algorand[21] combine two different consensus algorithms, both of which must be used to mine a single block.

Our hybrid PoW consensus enables original bitcoin ASIC miners to participate in mining without changing their hardware. Moreover, it enables the competition between solo miners (Now, Bitcoin mining is a competition between mining pools, but not the solo miners) in mining.

The rest of this paper is organized as follows. Section 2 introduces a non-outsourceable PoW puzzle and hybrid consensus. Section 3 presents the difficulty adjustment algorithm in hybrid consensus. Section 4 analyzes the security of proposed scheme. Section 5 provides the simulation results. Discussion is given in Section 6. Finally, we conclude with Section 7.

2. PROPOSED SCHEME

2.1. ECDSA

Let G be an Elliptic curve group of order q with generator point P. The private key is a random integer $d(0 < d < q)$ and the public key is $Q = d \cdot P$. H denotes a cryptographic hash function whose outputs have bit-length no more than $|q|$, where $|q|$ is the bit-length of q.

The ECDSA signing operation on a message m is defined as follows:

Step1. Choose a random integer $k(0 < k < q)$.
Step2. Compute $R = k \cdot P$. Let $R = (r_x, r_y)$.
Step3. Compute $r = r_x \bmod q$. If $r = 0$, go to Step2.
Step4. Compute $s = k^{-1}(H(m) + rd) \bmod q$. If $s = 0$, go to Step1.
Step5. Output (r, s)

The ECDSA verifying operation on a message m and signature (r, s) is defined as follows:

Step1. Verify that r and s are integers in the interval$[1, q-1]$. If any verification fails then return "Reject".
Step2. Compute $w = s^{-1} \bmod q, u_1 = w\, H(m) \bmod q$ and $u_2 = r\, w \bmod q$.
Step3. Compute $Z = u_1 \cdot P + u_2 \cdot Q$. If $Z = \infty$ then return "Reject". Let $Z = (z_x, z_y)$.
Step4. If $z_x \bmod q = r$ then return "Accept"; Else return "Reject".

2.2. PROPOSED NON-OUTSOURCEABLE PROOF OF WORK PUZZLE

Hash based Proof of Work puzzle involves finding a valid nonce $n \in N$ such that

$$H(m||n) \leq Target \tag{1}$$

where m is a block header value and $Target$ is a 256-bit value to determine the difficulty of mining. A miner repeatedly tries different values of n until Equation (1) is satisfied. The block header value m is the collection of inputs specific to a block and can be denoted as

$m = v||PrevBlockHash||MR(CB, TR_1, \cdots, TR_t)||T_s||Target$, where v is a (software) version number, $prevBlockHash$ is the hash value of the previous block, CB is a coinbase transaction for the miner who first publishes a valid block, $TR_1, \cdots, TR_t$ is a set of valid transactions not yet confirmed, $MR(x)$ denotes the root of the Merkle tree over transactions x and T_s is the time stamp.

We modified Equation (1) as follows.

$$H(m||n||PubKey_{CB}||HeaderSig(m||n, privKey_{CB})) \leq Target \tag{2}$$

Here, $HeaderSig(m||n, privKey_{CB})$ is a ECDSA signature for $m||n$, where $privKey_{CB}$ is the coinbase transaction(CB)'s private key corresponding to $PubKey_{CB}$ which produces the address of coinbase transaction. In our scheme, the coinbase transaction must include only one P2PKH address as a reward address. In block header, $PubKey_{CB}$ is used to verify the signature. Meanwhile, in the coinbase transaction, it is used to verify the receiving (P2PKH) address. This is a non-outsourceable puzzle[4, 5].

By defining

$$X = m||n||PubKey_{CB}||HeaderSig(m||n, privKey_{CB}) \tag{3}$$

, mining problem can be considered as the process to find X that satisfy $H(X) \leq Target$.

Miner can find a block by iterating through the nonce and extra nonce until the resulting hash is below the target. Meanwhile, miner can change X by repeatedly signing the fixed $m||n$(i.e., without changing nonce and extra nonce) because ECDSA signature utilizes randomization in the signature generation.

By substituting $HeaderSig(m||n, privKey_{CB}) = r||s$, $s = k^{-1}(H(m||n) + r\, privKey_{CB}) mod\ q$ and $m||n = M$ in Equation (3), following Equation (4) can be obtained.

$$X = M||PubKey_{CB}||r||(k^{-1}(H(M) + r\, privKey_{CB}) mod\ q \tag{4}$$

From Equation (4), it can be seen that it is possible to change X by altering $M = m||n$ for fixed r. In this case, signature process requires only one hash(i.e., $H(M)$) and one modular multiplication($k^{-1}H(M)\ mod\ q$). However, if r is changed for fixed M, signature process requires at least one elliptic curve point multiplication($k \cdot P$), one modular inverse ($k^{-1}\ mod\ q$) and two modular multiplication($k^{-1}H(M)\ mod\ q$ and $k^{-1}r\ privKey_{CB}\ mod\ q$).

From the facts above, miner would rather change only M through the nonce and extra nonce than repeatedly sign the fixed M for the many trials of X values in the given time.

Meanwhile, it is a well-known that for every valid signature (r, s), the pair $(r, -s)$ is also a valid signature in EC-DSA. In order to make (r, s) unique, Step5 of EC-DSA signing can be modified as follows.

Step5. If $s \le (q-1)/2$ then outputs (r, s); Else outputs $(r, q-s)$.

In this case, Step1 of verifying of EC-DSA is modified as follows.

Step1. Verify that $1 \le r \le q-1$ and $1 \le s \le (q-1)/2$. If any verification fails then return "Reject".

Modified EC-DSA as above or Schnorr signature[22] which is known to be strongly secure can be used in our scheme (more precisely, in Equation(2)).

Compared to PoW algorithm of Ergo [8], our non-outsourceable Pow algorithm is not resistant to ASICs and by this property, our hybrid consensus protocol becomes to be secure(mentioned in Section 4).

Even our non-outsourceable scheme is based on the application of a digital signature, only one modular multiplication($k^{-1}H(M)\ mod\ q$) is added to original bitcoin Pow calculation. This gives the possibility to make the ASIC that can be used to solve both non-outsourceable and outsourceable Pow puzzle. To remove this probability, in our non-outsourceable puzzle, KECCAK-256[23] is used instead of SHA-256 as a hash function in Equation(2) and Merkle root generation.

2.3.HYBRID CONSENSUS

Bitcoin PoW involves finding a valid solution n to the following problem:

$$SHA - 256^2(m||n) \le Target_{SHA} \tag{5}$$

Meanwhile, our non-outsourceable Pow involves finding a valid solution n to the following problem.

$$KECCAK - 256^2(m||n||PubKey_{CB}||HeaderSig(m||n, privKey_{CB})) \le Target_{KECCAK} \tag{6}$$

We let $H_1(\cdot)$ represents the PoW function of original bitcoin and $H_2(\cdot)$ represents the PoW function of our non-outsourceable scheme. Let $T_1 = Target_{SHA}$ and $T_2 = Target_{KECCAK}$.

Then, Equation(5) and (6) can be described as follows.

$$H_1(m||n) \le T_1 \tag{7}$$

$$H_2(m||n) \le T_2 \tag{8}$$

In our hybrid consensus protocol, miner can generate the block by solving original bitcoin puzzle(i.e., Equation(7) : Puzzle1) or our non-outsourceable puzzle(i.e., Equation(8) : Puzzle2).

If our hybrid consensus protocol is applied to starting from the $k\text{-}th$ block generation, the blockchain can be denoted as

$$B_0 \leftarrow B_1 \leftarrow \cdots \leftarrow B_{k-1} \leftarrow T_k \leftarrow T_{k+1} \leftarrow T_{k+2} \leftarrow$$

where $B_j (\in B: 0 \le j < k)$ is original bitcoin block and $T_j (\in B$ or $C: j \ge k)$ is block generated by our scheme. In this case, B is the set of blocks generated by Puzzle1 and C is the set of blocks generated by Puzzle2.

Our scheme requires a little addition to the way current client software interprets the information stored in the blockchain because of the blocks generated by our non-outsourceable puzzle(Table I).

3. DIFFICULTY ADJUSTMENT

3.1. DIFFICULTY ADJUSTMENT IN BITCOIN

The block creation rate λ is given by

$$\lambda = \frac{R}{D} \tag{9}$$

where R denotes the hash rate and D denotes the difficulty.

Let t_0 and t be the desired time and actual time consumed to create $N(= 2016)$ blocks respectively. Then,

$$\lambda = \frac{N}{t} \tag{10}$$

and

$$\lambda_0 = \frac{N}{t_0} \tag{11}$$

are satisfied where λ_0 is the desired block creation rate.

TABLE I HEADER OF BLOCK GENERATED BY PUZZLE2

Value		Description
Meta data	Version	(4 bytes) protocolversion.
	Previous block	(32 bytes) the hash (twice SHA256 or twice KECCAK256) of the header of the previous block.
	Merkle root	(32 bytes) the hash (KECCAK 256) of the root of the Merkle tree that summarizes all the transactions in the block.
	Timestamp	(4 bytes) approximate creation time of the block. (Unix epoch)
	Difficulty target	(4 bytes) difficulty target for the block.
	Nonce	(4 bytes) the nonce used for the proof-of-work
Public key		(64 byte) public key corresponding to coinbase reward address
Signature of Meta data		(64 byte) the signature of Meta data, which uses private key that matches the Public key

From Equation(9), (10) and (11),

$$\lambda_0 t_0 = \lambda t \tag{12}$$

and

$$\lambda^+ = \frac{R}{D^+} = \lambda_0 = \lambda \cdot \frac{t}{t_0} = \frac{R}{D} \cdot \frac{t}{t_0} \tag{13}$$

where D^+ and λ^+ are the difficulty and block creation rate which are adapted in the next round.

From Equation(13),

$$D^+ = D \cdot \frac{t_0}{t} \tag{14}$$

Equation(14) ensures that the block creation rate is stable even if the hash rate is changed.

3.2. DIFFICULTY ADJUSTMENT IN HYBRID CONSENSUS

Let us denote difficulty, hash rate and block creation rate of Puzzle1 by D_1, R_1, λ_1 and those of Puzzle2 by D_2, R_2 and λ_2. Assume that $N_1$and N_2 are the numbers of blocks generated by Puzzle1 and Puzzle2 in time t and $N_1 + N_2 = N = 2016$.

From the definition of block creation rate, following Equations are satisfied

$$N_1 = \lambda_1 t \tag{15}$$

$$N_2 = \lambda_2 t \tag{16}$$

$$N = \lambda t \tag{17}$$

In this case, λ is the total block creation rate in hybrid consensus. From this,

$$\lambda t = N = N_1 + N_2 = \lambda_1 t + \lambda_2 t = (\lambda_1 + \lambda_2)t$$

Thus, we can derive the relation between λ, λ_1 and λ_2 as

$$\lambda = \lambda_1 + \lambda_2 \tag{18}$$

On the other hand, from Equation(9), we have $\lambda_1 = \frac{R_1}{D_1}$ and $\lambda_2 = \frac{R_2}{D_2}$.

Assume that

$$\mu = \frac{D_1}{D_2} \tag{19}$$

Then, Equation(18) can be written as follows.

$$\lambda = \lambda_1 + \lambda_2 = \frac{R_1}{D_1} + \frac{R_2}{D_2} = \frac{R_1}{D_1} + \frac{\mu R_2}{D_1} = \frac{R_1 + \mu R_2}{D_1} \tag{20}$$

Equation(20) shows that it is always possible to adjust $\lambda_2 = \frac{R_2}{D_2} = \frac{\mu R_2}{D_1}$ so that it is equal to $\lambda_1 = \frac{R_1}{D_1}$ by regulating $\mu = \frac{D_1}{D_2}$, even if R_1 and R_2 are different from each other.

We will now show the method to control the difficulties D_1 and D_2 so that $N_1 = N_2$(i.e.,$\lambda_1 = \lambda_2$) and $N_1 + N_2 = N$(i.e., total block creation rate $\lambda(= \lambda_1 + \lambda_2 = 2\lambda_1 = 2\lambda_2)$ matches a desired value $\lambda_0 = N/t_0$).

From the assumption, $N_1 \neq N_2$(i.e.,$\lambda_1 \neq \lambda_2$) and $N = N_1 + N_2 (= 2016)$ blocks are generated in time $t (\neq t_0)$.

First, we make the number of blocks generated by Puzzle2 equal to N_1 by regulating the difficulty of Puzzle2. In other words, we set $\mu^+ = \frac{D_1}{{D_2}^*}$ so that $\lambda_1 = {\lambda_2}^*$ where ${\lambda_2}^*$ is the block creation rate of Puzzle2 corresponding to ${D_2}^*$, which is the updated difficulty of Puzzle2.

From Equation(15) and (16), $\frac{N_1}{\lambda_1} = \frac{N_2}{\lambda_2}$ and $\lambda_1 = \lambda_2 \cdot \frac{N_2}{N_1}$.

From the definition of block creation rate(i.e., ${\lambda_2}^* = \frac{R_2}{{D_2}^*}$),

$$\lambda_1 = {\lambda_2}^* = \frac{R_2}{{D_2}^*} = \frac{R_2}{D_2} \cdot \frac{N_1}{N_2}.$$

Hence,

$${D_2}^* = D_2 \cdot \frac{N_2}{N_1} \tag{21}$$

From Equation(19) and (21),

$${D_2}^* = \frac{D_1}{\mu^+} = D_2 \cdot \frac{N_2}{N_1} = D_1 \cdot \frac{N_2}{\mu N_1}$$

and so,

$$\mu^+ = \mu \cdot \frac{N_1}{N_2} \tag{22}$$

By using μ^+, we have

$$\lambda_1 = {\lambda_2}^* = \frac{R_1}{D_1}$$

and

$$\lambda^* = \lambda_1 + {\lambda_2}^* = 2\lambda_1 = \frac{2R_1}{D_1} \tag{23}$$

where λ^* is the total block creation rate after difficulty update (i.e., ${D_2}^* = \frac{D_1}{\mu^+}$). By updating the difficulty of Puzzle2, $2N_1$ blocks would be mined in time t because N_1 blocks are generated in Puzzle2, too. From this,

$$\lambda^* = \frac{2N_1}{t} \tag{24}$$

Second, we make λ^* equal to λ_0 by regulating the difficulty of Puzzle1 in Equation(23). In other words, we set ${D_1}^+$ so that total block creation rate $\lambda^+ (= {\lambda_1}^+ + {\lambda_2}^+ = \frac{R_1}{{D_1}^+} + \frac{R_2}{{D_2}^+} = 2{\lambda_1}^+ = \frac{2R_1}{{D_1}^+})$ matches a desired value λ_0(i.e., $\lambda^+ = \lambda_0$) where ${D_1}^+$, ${D_2}^+$, ${\lambda_1}^+$ and ${\lambda_2}^+$are the difficulties and block creation rates of Puzzle1 and 2 which are adapted in the next round.

From Equation(11), λ^+ may be written as follows.

$$\lambda^+ = \lambda_0 = \frac{N}{t_0} = \frac{N_1 + N_2}{t_0} \tag{25}$$

Let $\beta = \frac{\lambda_0}{\lambda^*}$. From Equation(24) and (25),

$$\beta = \frac{\lambda_0}{\lambda^*} = \frac{N_1 + N_2}{2N_1} \cdot \frac{t}{t_0} \tag{26}$$

and

$$\lambda^+ = \beta \cdot \lambda^* \tag{27}$$

are satisfied.

From Equation(23) and (27),

$$\lambda^+ = \frac{2R_1}{{D_1}^+} = \frac{N_1 + N_2}{2N_1} \cdot \frac{t}{t_0} \cdot \frac{2R_1}{D_1} \tag{28}$$

and as a result,

$${D_1}^+ = D_1 \cdot \frac{2N_1}{N_1 + N_2} \cdot \frac{t_0}{t} = D_1 \cdot \frac{N_1}{N/2} \cdot \frac{t_0}{t} \tag{29}$$

is satisfied.

And from Equation(22) and (29),

$${D_2}^+ = \frac{{D_1}^+}{\mu^+} = D_2 \cdot \frac{2N_2}{N_1 + N_2} \cdot \frac{t_0}{t} = D_2 \cdot \frac{N_2}{N/2} \cdot \frac{t_0}{t} \tag{30}$$

From the facts above, difficulty retargeting algorithm of our hybrid scheme can be described as follows.

Let ${N_1}^+$ and ${N_2}^+$ be the numbers of blocks generated by Puzzle1 and Puzzle2 of next round in time t_0.

In order to satisfy ${N_1}^+ = {N_2}^+$(i.e., ${\lambda_1}^+ = {\lambda_2}^+$) and ${N_1}^+ + {N_2}^+ = N = 2016$(i.e., ${\lambda_1}^+ + {\lambda_2}^+ = \lambda_0$), ${D_1}^+, {D_2}^+$ and μ^+ have to be regulated as follows.

Algorithm 1 Difficulty retargeting in hybrid consensus

Input: $N_1, N_2 (N_1 + N_2 = N, N_1 > 0, N_2 > 0), D_1, D_2, t$

Step1: $\mu = \frac{D_1}{D_2}$

Step2: $\mu^{+} = \mu \cdot \frac{N_1}{N_2}$

Step3: ${D_1}^{+} = D_1 \cdot \frac{2N_1}{N_1+N_2} \cdot \frac{t_0}{t}$

Step4: ${D_2}^{+} = \frac{{D_1}^{+}}{\mu^{+}}$

Output: ${D_1}^{+}, {D_2}^{+}$

Such a regulation ensures that in Puzzle1 and 2, the block creation rates are equal even if the hash rates are not. Besides, it ensures that total block creation rate is stable in hybrid consensus. Equation(29) and (30) show that D_1 and D_2 are adjusted independently so that both Puzzle1 and 2 generate $N/2$ blocks, respectively, in time t_0(i.e., ${\lambda_1}^{+} = {\lambda_2}^{+} = \lambda_0/2$). Hence, it is possible to describe Algorithm1 by using only Equation(29) and (30). However, we used μ and μ^{+} in the description of Algorithm1 because μ is used in the security analysis and simulation.

Note. We assume that hash rates $R,\ R_1$ and R_2 are constant over each single difficulty adjustment interval. This is reasonable when considered as average of hash rate.

4. SECURITY ANALYSIS

4.1. SECURITY ANALYSIS OF BITCOIN

Let R_1 be the total hash rate of Puzzle1. Then, R_1 can be written as

$$R_1 = \sum_{i=1}^{\eta} R_{1i} \tag{31}$$

, where η is the number of pools, R_{1i} is the hash rate of the i-th pool and $R_{1i} > R_{1i+1}$ for all $i(0 < i < \eta)$.

When $\alpha_1 = \frac{R_{11}}{R_1} > 0.5$, 51% attack is possible. In current bitcoin mining, $\alpha_2 = \frac{R_{1sub}}{R_1} > 0.5$ is satisfied for $\pi = 4{\sim}5$, when

$$R_{1sub} = \sum_{i=1}^{\pi} R_{1i} \tag{32}$$

In other words, a few pool operators can control more than 51% of the total network's hashing power in bitcoin. Of course, such problems have already been known, no practical solutions have been implemented in bitcoin.

Note. From Equation (31) and (32), α_1 and α_2 which are discussed in all attacks to bitcoin's consensus can be in the interval $(0,1]$.

4.2. SECURITY ANALYSIS OF PROPOSED SCHEME

From Equation(19) and (20), proposed hybrid consensus can be seen as single puzzle with difficulty D_1 and hash rate $R = R_1 + \mu R_2$. And from Section 3, $R_1 = \mu R_2$ is satisified and so, $R = 2R_1$.

Hence, from Equation (31) and (32),

$$\alpha_1 = \frac{R_{11}}{R} = \frac{R_{11}}{2R_1} < \frac{R_1}{2R_1} = 0.5 \tag{33}$$

and

$$\alpha_2 = \frac{R_{1sub}}{R} = \frac{R_{1sub}}{2R_1} < \frac{R_1}{2R_1} = 0.5 \tag{34}$$

From Equation(33) and (34), both α_1 and α_2 are restricted to being in the interval $(0,1/2]$. From this, it can be seen that in our hybrid consensus, a few pool operators cannot control more than 51% of the total network's hashing power.

As mentioned above, proposed scheme seems to be double puzzle consensus, but it is essentially bitcoin. Our scheme only increases the total hashing power of the bitcoin network by adding the non-outsourceable hashing power which cannot participate in pool formation. Such increase of total hashing power would arguably weaken the pool's forces. In other words, there is no possible way for a single pool or combination of a few pools to control more than 51 percentage of the total mining power and all known attacks caused by pool such as selfish mining[24-29] and block withholding[30] could be weakened.

4.3. INITIAL DIFFICULTY ADJUSTMENT FOR THE SECURITY

R_{1i} of Equation(31) may be expressed as

$$R_{1i} = A_{1i} + B_{1i}$$

where A_{1i} is the hash rate of non-programmable mining hardware(e.g., ASIC) and B_{1i} is that of programmable mining hardware(e.g., GPU).

It is obvious that A_{1i} is much larger than B_{1i}; however, A_{1i} will never be able to participate in solving Puzzle2, and only B_{1i} might try to solve Puzzle2.

Now, assume that

$$B_{1j} = max\{B_{1i} | i = \overline{1,\eta}\}$$

From the target regulation of the Section 3, it can be shown that

$$\lambda_1 = \frac{R_1}{D_1} = \lambda_2 = \frac{R_2}{D_2}$$

and the total block creation rate λ can be denoted as follows.

$$\lambda = \lambda_1 + \lambda_2 = \frac{2R_2}{D_2}$$

And R_2 may be expressed as

$$R_2 = A_2 + B_2$$

where A_2 is the hash rate of non-programmable mining hardware and B_2 is the hash rate of programmable mining hardware. It is trivial that A_2 is much greater than B_2 because Puzzle2 is not designed to be ASIC resistant.

Note that Ethereum uses an ASIC resistant PoW function [31].

If $B_{1j} > 2R_2$(Of course, it would be possible only in the initial stage, when our hybrid consensus is first applied to bitcoin.), then j-th mining pool of Puzzle1 will have a potentiality to cause the 51% attack by solving Puzzle2 since the following equation is satisfied:

$$\lambda_{2j} = \frac{B_{1j}}{D_2} > \frac{2R_2}{D_2} = 2\lambda_2 = \lambda$$

Puzzle2 is non-outsourceable and B_{1j} is the hash rate of GPU pool. Hence, attack to Puzzle2 by B_{1j} is theoretically impossible. However, j-th mining pool has the possibility to generate the longest chain by using B_{1j} and R_{1j}(i.e., possibility of selfish mining).

In order to prevent the creation of Puzzle2 blocks by GPU pools of Puzzle1, we should not only allow Puzzle2 ASICs, but also encourage them to participate in Puzzle2 mining. Hence, the Puzzle2 must have no resistance to ASIC.

Suppose that

$$2A_2 > B_{1j} \tag{35}$$

Then, it is self-evident that $2R_2 > B_{1j}$ and this shows that Puzzle 2 must encourage A_2 to increase rapidly so that it overpowers the hash rate of the massive GPU pool(i.e., B_{1j}).

Of course, at first, B_{1j} might be remarkably greater than A_2. However, when considering the fact that a small pool of the latest ASIC miners can surpass the large pool of GPU miners in speed, A_2 will become larger than B_{1j} within a short period of time and if Equation(35) is once satisfied, there would be no possibility of 51% attack by B_{1j}.

From the facts above, while $R_2 < B_{1i}$ is satisfied (i.e., initial stage of hybrid consensus), target regulation must ensure that $\lambda_1 = \frac{R_1}{D_1} > \lambda_{2j} = \frac{B_{1j}}{D_2}$. In this case, $\lambda_2 = \frac{R_2}{D_2} < \frac{B_{1j}}{D_2} = \lambda_{2j}$ and so, $N_1 > N_2$ would be satisfied. In other words, Algorithm 1 of Section 3 cannot be used in initial stage of hybrid consensus. Hence, the initial difficulty adjustment would have to be considered. Let A_1 and B_1be the hash rate of non-programmable and programmable mining hardware, respectively, in Puzzle1. Then, $R_1 = A_1 + B_1$.

Assume that the relation between R_1 and B_1 can be approximately estimated. In other words, ε=$\frac{R_1}{B_1}$ and $R_1 = \varepsilon B_1 > \varepsilon B_{1j}$. Then, D_2 must be set to $\frac{D_1}{\varepsilon}$ so that $\frac{R_1}{D_1} = \frac{B_1}{D_2} > \frac{B_{1j}}{D_2}$. In this case, total block creation rate λ can be denoted as follows.

$$\lambda = \lambda_1 + \lambda_2 = \frac{R_1}{D_1} + \frac{R_2}{D_2} = \frac{R_1 + \varepsilon R_2}{D_1}$$

And from Equation(12),

$$\lambda^+ = \frac{R_1}{{D_1}^+} + \frac{R_2}{{D_2}^+} = \frac{R_1 + \varepsilon R_2}{{D_1}^+} = \lambda_0 = \lambda \cdot \frac{t}{t_0} = \frac{R_1 + \varepsilon R_2}{D_1} \cdot \frac{t}{t_0} \tag{36}$$

where ${D_1}^+, {D_2}^+$ and λ^+ are the difficulty of Puzzles and block creation rate which are adapted in the next round.

From Equation(36),

$${D_1}^+ = D_1 \cdot \frac{t_0}{t}$$

and

$${D_2}^+ = \frac{{D_1}^+}{\varepsilon} = D_2 \cdot \frac{t_0}{t}$$

Let ${N_1}^+$ and ${N_2}^+$ be the numbers of blocks generated by Puzzle1 and Puzzle2 of next round in time t_0.

In order to satisfy ${N_1}^+ + {N_2}^+ = N = 2016$(i.e., ${\lambda_1}^+ + {\lambda_2}^+ = \lambda_0$), ${D_1}^+$ and ${D_2}^+$ have to be regulated as follows.

```
Algorithm 2 Difficulty retargeting in initial stage of hybrid consensus
Input: $N_1, N_2 (N_1 + N_2 = N, N_1 > 0), D_1, D_2, t, \varepsilon > 0$
Step1: $\tau = \frac{N_2}{N_1}$. If $\tau > 1$ then go to Step1 of Algorithm1
Step2: ${D_1}^+ = D_1 \cdot \frac{t_0}{t}$
Step3: ${D_2}^+ = \frac{{D_1}^+}{\varepsilon}$
Output: ${D_1}^+, {D_2}^+$
```

Such a regulation ensures that total block creation rate is stable and our hybrid consensus is secure from 51% attack in the initial stage. We assume that $\frac{R_1}{\varepsilon} > B_{1j}$ is satisfied during the whole period of transition(i.e., $\tau \le 1$). This is reasonable because, in current bitcoin, A_1 is much larger than B_1 (i.e., $R_1 = \varepsilon B_1 > \varepsilon B_{1j}$) and disparity between A_1 and B_1 goes on increasing(i.e., $\frac{R_1}{B_1}$ becomes larger than ε) as time advances(See Figure1.(a) of Section 5.). If $\tau > 1$(i.e, $R_2 > B_{1j}$) is once satisfied, then $A_1 > A_2 > B_1 > B_2$ would be satisfied and Algorithm1 would be used instead of Algorithm2. The only difference between Algorithm1 and 2 is the way that $\frac{D_1}{D_2}$ is chosen.

5. SIMULATION RESULT

We evaluated our hybrid consensus scheme with 1000-node experiments on the emulated network. We denote a round as the difficulty retarget period of 2016 blocks. Hence, PoW difficulty is constant in a round (i.e., difficulty is not adjusted for 2016 blocks) and is adjusted dynamically when the round is switched.

In practice, the hash rate of total network is continuously changed during a single round, but it is constant during the short period of time from any given moment even if round is being switched. However, in simulation, we assumed that the hash rate is constant during the whole period of a single round and is changed only when the round is switched. Of course, this does not contravene the practice when considered as the average hash rate of the round(mentioned in Section 3).

For simplicity, we set $\varepsilon = 6$ and measured N_1/N_2, t_0/t and u changing R_1 and R_2(more precisely, R_1/R_2). Figure1 shows the simulation result of our hybrid scheme. B^* denotes the B_1 of middle round(i.e., Round 15). As can be seen, real crossover point at which Algorithm2 (u is equal to ε) is replaced by Algorithm1(u is close to R_1/R_2 of previous round) is in round 12. And, by Algorithm1, N_1/N_2 and t_0/t are kept close to 1. On the other hand, the ideal crossover point at which εR_2 becomes larger than R_1 is in Round 9. Thus, it would be possible to replace Algorithm2 by Algoritm1 in Round 10. However, unlike simulation, R_1 and R_2 are not known(only N_1 and N_2 are known) in practice, so the real point at which N_2 becomes larger than N_1 is used instead of ideal point. From lottery property of blockchain mining, the real point can be placed before the ideal point, but it can be ignored when considered the fact that both are placed after the point at which R_2 becomes larger than B_1(i.e., Round3). Overall, simulation result shows the possibility of seamless transition from original bitcoin to our hybrid scheme. In simulation, we changed R_1 and R_2 drastically(e.g., Round 15, 16), but Algorithm1 ensured the smooth regulation. However, in the real world, such sudden changes of R_1 and R_2 which are the average hash rates of a round(approximately 2 weeks) are not possible.

Note. Recently, bitcoin hash rate, which is the average hash rate of retargeting interval, has never increased more than two times (or decreased less than one half) within a period of 2 weeks. This can be seen from the analysis of the bitcoin hash rate over a period of four or five years.

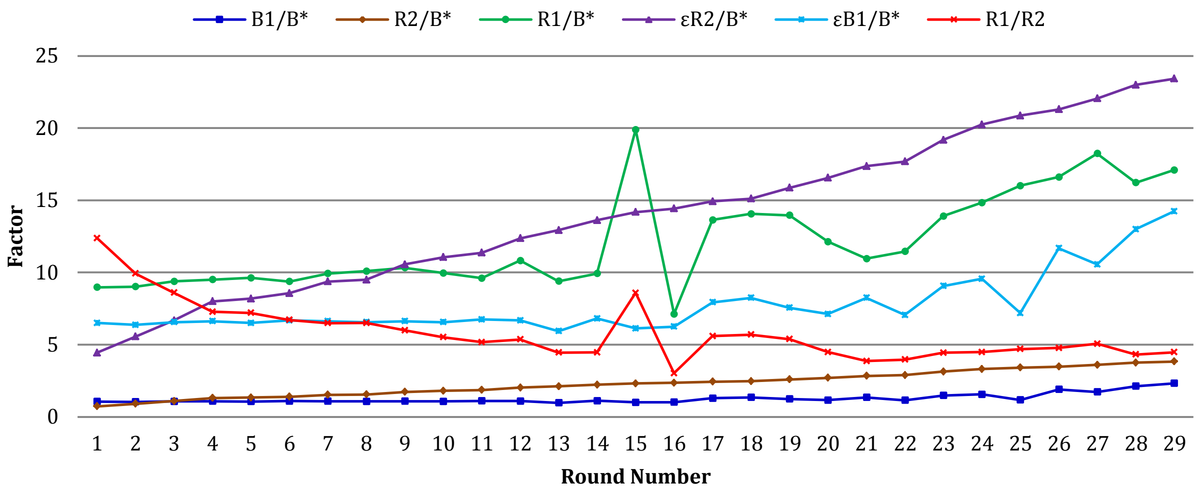
6. DISCUSSION

Until now, we considered only the case of $\theta \left(= \frac{N_2}{N_1}\right) = 1$. However, different values of θ can be used in our hybrid scheme. In other words, our solution is highly tunable. If θ is small (e.g., $\theta < 0.02$), then hybrid PoW becomes outsourceable PoW as the original Bitcoin. If θ is large (e.g., $\theta > 50$), then hybrid PoW becomes non-outsourceable PoW which uses Puzzle2. With the increase of θ, the work of our hybrid consensus can be smoothly shifted from outsourceable PoW to non-outsourceable PoW.

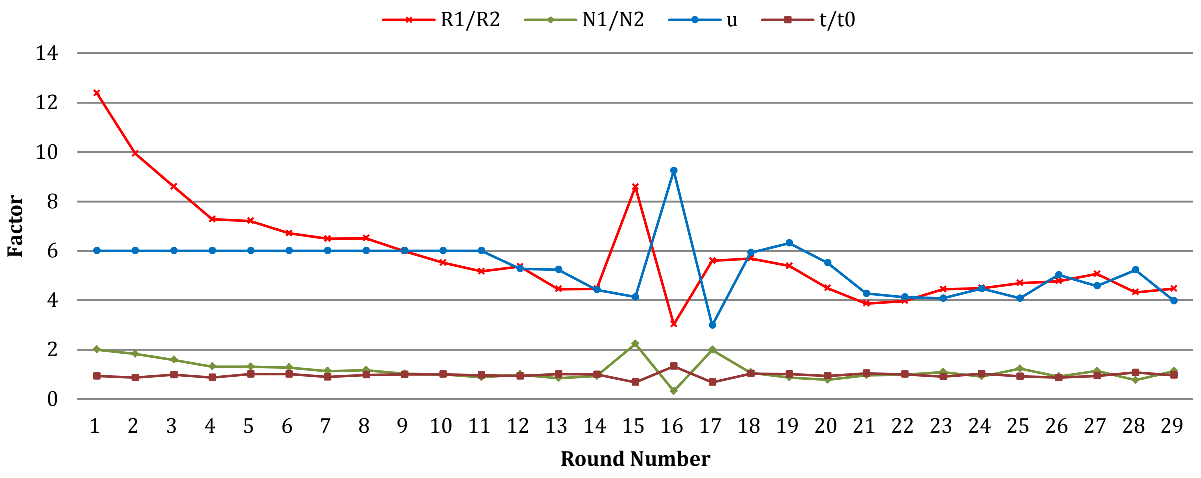
When $\theta \neq 1$, difficulty adjustment can be described as follows.

$$\mu^+ = \mu \cdot \frac{N_1}{N_2} \cdot \theta \tag{37}$$

$$ {D_1}^+ = D_1 \cdot \frac{N_1(1+\theta)}{N_1+N_2} \cdot \frac{t_0}{t} \tag{38}$$



(a)



(b)

Figure1. Simulation results of our scheme. (a) : Input parameters. (b) : Output parameters.

$$D_2{}^+ = \frac{D_1{}^+}{\mu^+} = D_2 \cdot \frac{N_2(1+\theta)}{(N_1+N_2)\theta} \cdot \frac{t_0}{t} \tag{39}$$

Equation(37) and (38) are identical to (22) and (29), respectively, for $\theta = 1$. When $\theta \neq 1$, Equation (33) and (34) can be described as follows.

$$\alpha_1 = \frac{R_{11}}{R} = \frac{R_{11}}{(1+\theta)R_1} < \frac{R_1}{(1+\theta)R_1} = \frac{1}{(1+\theta)}$$
$$\alpha_2 = \frac{R_{1sub}}{R} = \frac{R_{1sub}}{(1+\theta)R_1} < \frac{R_1}{(1+\theta)R_1} = \frac{1}{(1+\theta)}$$

It is an open problem to find an optimum value of θ at which the hybrid consensus system is efficient and secure from all known attacks.

Following facts would add the incentive for the miners to participate in Puzzle2 mining.

First, similar to non-outsourceable Puzzle of [8], Puzzle2 mining seems to be a race of signing power, but it is essentially the race of hashing power. Hence, it is not difficult to build the mining hardware for Puzzle2.

Second, R_1 is usually much larger than R_2 and from $\frac{R_1}{D_1} = \frac{R_2}{D_2}$, D_2 is much smaller than D_1. Hence, it would be well advised to solve the puzzle with difficulty D_2.

Of course, it still remains an open problem that Puzzle2 solo miners would actually appear in our hybrid scheme. However, the only clear thing is that regardless of appearance of Puzzle2 solo miners, in hybrid consensus, extant bitcoin pool miners will still be able to continue Puzzle1 mining by using hardware such as ASIC and GPU which they have used. When Puzzle2 solo miners appear, the only impact on extant Puzzle1 mining is that revenue gain of every miner of Puzzle1 is reduced by a factor of $1/(1+\theta)$.

Unlike all previous works, in our hybrid PoW scheme, both pool miners and solo miners could have an practical opportunity for block generation through Puzzle1 and 2, respectively.

7. CONCLUSION

In this paper, we have presented a hybrid PoW consensus protocol in order to discourage centralization and tackle the 51% attack. In the proposed scheme, miner can generate the block by solving either original outsourceable bitcoin puzzle or our non-outsourceable puzzle.

The main feature of our scheme is that it is fully compatible with current bitcoin designs, i.e., it can be implemented right now, because it preserves both the existing blockchain and investments which have been made in mining hardware(Section 2.3).

Especially, our scheme gives a possibility of seamless transition from the existing bitcoin system to new one (Section 3.2, Section 4.3 and Section 5) and a possibility that tunes tradeoff between outsourceable and non-outsourceable mining(Section 6). In contrast to the current bitcoin system and any other preceding protocols, our scheme presented two puzzles, but still is single-tired.

Finally, other puzzles which are known to be non- outsourceable could be used with original bitcoin puzzle in our hybrid consensus protocol.

REFERENCES

[1] Adam Back. Hashcash - a denial of service counter-measure. http://www.hashcash.org/papers/hashcash.pdf, 2002.
[2] S.Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf
[3] Ethereum Foundation. Ethereum’s white paper. https://github:com/ethereum/wiki/wiki/White-Paper, 2014.
[4] I.Eyal, E.G.Sirer, “How to disincentivize large bitcoin mining pools,” Blog post: http://hackingdistributed.com/2014/06/18/how-to-disincentivize-large-bitcoin-mining-pools/, June 2014.
[5] ziftrCOIN: a cryptocurrency to enable commerces. (2014). https://d19y4lldx7po3t.cloudfront.net/assets/docs/ziftrcoin-whitepaper-120614.pdf. Accessed 05 Nov 2016
[6] A. Miller, A. Kosba, J. Katz, and E. Shi, “Nonoutsourceable scratch-off puzzles to discourage bitcoin mining coalitions,” in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communication Security. ACM, 2015, pp. 680-691.
[7] P.Daian, I.Eyal and A.Juels, Piecework: Generalized outsourcing control for proofs of work, In International Conference on Financial Cryptography and Data Security, pages 182-190. Springer, 2017.
[8] A. Chepurnoy, V. Kharin, and D. Meshkov, “Autolykos: The ergo platform pow puzzle,” 2019. [Online]. Available: https://docs.ergoplatform.com/ErgoPow.pdf
[9] Ergo Developers. Ergo: A resilient platform for contractual money. https://ergoplatform.org/docs/whitepaper.pdf, 2019.
[10] L. Luu, Y. Velner, J. Teutsch, and P. Saxena, “SmartPool: Practical decentralized pooled mining,” in Proc. 26th USENIX Security Symposium, 2017, pp. 1409–1426
[11] Ethash. [Online]. Available: https://github.com/ethereum/wiki/wiki/Ethash/6e97c9cea49605264c6f4d1dc9e1939b1f89a5a3
[12] A. Biryukov and D. Khovratovich, “Equihash: Asymmetric proof-of-work based on the generalized birthday problem,” Ledger, vol. 2, pp. 1–30, 2017.
[13] I. Bentov, R. Pass, and E. Shi, “Snow white: Provably secure proofs of stake,” IACR Cryptology ePrint Archive, vol. 2016, p. 919, Sep. 2016
[14] I. Bentov, C. Lee, A. Mizrahi, and M. Rosenfeld, “Proof of activity: Extending bitcoin’s proof of work via proof of stake (extended abstract),” ACM SIGMETRICS Performance Evaluation Review, vol. 42, on. 3, pp. 34-37, Dec. 2014.
[15] V. Buterin and V. Griffith, “Casper the friendly finality gadget,” arXiv preprint arXiv:1710.09437, 2017.
[16] C. Decker, J. Seidel, and R. Wattenhofer, “Bitcoin meets strong consistency,” in Proceedings of the 17th International Conference on Distributed Computing and Networking, ser. ICDCN’16, Singapore, 2016, pp. 13:1-13:10.
[17] R. Pass and E. Shi, “Hybrid Consensus: Efficient Consensus in the Permissionless Model,” in 31st International Symposium on Distributed Computing (DISC 2017), vol. 91, Vienna, Austria, Oct. 2017, pp. 39:1-39:16.
[18] J. Kwon, “Tendermint: Consensus without mining (draft),” Self-published Paper, fall 2014. [Online], Available: https://tendermint.com/static/docs/tendermint.pdf
[19] “Proof of authority chains,” Jan. 2018. [Online]. Available: https://github.com/paritytech/parity
[20] D. Larimer, “Delegated proof-of-stake (dpos),” Bitshare whitepaper, Tech. Rep., 2014.
[21] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, “Algorand: Scaling byzantine agreements for cryptocurrencies,”in Proceedings of the 26th Symposium on Operating Systems Principles (SOSP’17). Shanghai, China: ACM, Oct. 2017, pp. 51-68.
[22] C.P.Schnorr, Efficient signature generation by smart cards. Journal of Cryptology, 4(3):161-174, 1991.
[23] G. Bertoni, J. Daemen, M. Peeters, G. van Assche, R. van Keer, "Keccak implementation overview", 2012, Available: https://keccak.team/files/Keccak-implementation-3.2.pdf
[24] I.Eyal, E. G. Sirer, “Majority Is Not Enough: Bitcoin Mining Is Vulnerable” in Proceedings of International Conference on Financial Cryptography and Data Security, Berlin, Heidelberg, 436-454, 2014
[25] K.Nayak, S.Kumar, A.Miller, and E.Shi, "Stuboorm mining: Generalizing selfish mining and combining with an eclipse attack" in 2016 IEEE European Symposium on Security and Privacy (EuroS P), Saarbrucken, Germany, MAr. 2016, 00. 305-320.
[26] M. Carlsten, "The impact of transaction fees on bitcoin mining strategies," Master's thesis, Princeton University, 2016.
[27] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in Financial Cryptography and Data Security: 20th International Conference, Revised Selected Paper, Christ Church, Barbados, Feb. 2017, pp. 515-532.
[28] J. Gobel, H. Keeler, A.Krzesinski, and P.Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay," Performance Evaluation, vol.104, no. Supplement C.pp. 23-41, 2016.
[29] J. Beccuti and C. Jaag, "The bitcoin mining game: On the optimality of honesty in proof-of-work consensus mechanism," Swiss Economics, Working Papers 0060. Aug. 2017. [Online]. Available: https://ideas.repec.org/p/chc/wpaper/0060.html

[30] A. Laszka, B. Johnson, and J. Grossklags, “When bitcoin mining pools run dry,” In Financial Cryptography and Data Security: FC 2015 International Workshops on BITCOIN. WAHC and Wearable, San Juan. Puerto Rico, Jan. 2015, pp. 63-77.

[31] Ethereum Foundation. Ethash proof of work. https://github:com/ethereum/wiki/wiki/Ethash.