electronics

MDPI

Article

Adjusting the Block Interval in PoW Consensus by Block Interval Process Improvement

Heesang Kim and Dohoon Kim *

Department of Computer Science, Kyonggi University, Suwon 16227, Gyeonggi-do, Korea; skewed@kyonggi.ac.kr

* Correspondence: karmy01@kyonggi.ac.kr

Abstract: Blockchain is not widely applied in various fields due to the critical issue of scalability as part of the blockchain trilemma. This issue arises during consensus among the nodes in a public blockchain. To address the issue of low scalability with proof-of-work (PoW) consensus, various methods have been proposed for transaction per second (TPS) improvement. However, no such methods include an improvement in the consensus step. Therefore, to improve PoW public blockchain scalability, it is important to shorten the time required for PoW consensus. This paper proposes a method for minimizing the block intervals that occur during consensus over a PoW blockchain network. A shortened block interval leads to an increase in the probability of three different attacks: selfish mining, double-spending, and eclipse attacks. According to an experiment using Ethereum, with a typical PoW blockchain, it is inevitable to provide rewards for stable block mining in competition between mining pools. To find an optimal block interval in the PoW consensus algorithm, we conducted a four-step experiment. The purpose of this experiment was to verify the difficulty level and issues with Mainnet security. Therefore, considering stale block mining rewards, an optimal block interval is proposed. The Ethereum TPS was improved by at least 200%. Given this finding, it is considered possible to achieve a similar improvement in a different PoW blockchain. On balance, even if the block interval is shorter than that of the PoW Mainnet, network security falls by only 1.21% in Testnet, even with a rise in the stale block rate, while performance is increased at up to 120 TPS, which is three times higher than that in Mainnet.

Keywords: blockchain; PoW; scalability; blockchain trilemma; consensus; block interval time; distributed network

Citation: Kim, H.; Kim, D. Adjusting the Block Interval in PoW Consensus by Block Interval Process Improvement. *Electronics* **2021**, *10*, 2135. https://doi.org/10.3390/electronics10172135

Academic Editor: Priyadarsi Nanda

Received: 31 July 2021
Accepted: 30 August 2021
Published: 2 September 2021

1. Introduction

Blockchain, a type of distributed ledger technology, is a distributed data processing technology. As the number of network nodes has exponentially increased and the communication speed between nodes increases, peer-to-peer (P2P) networks have become more decentralized. As a result, all the nodes participating in the network can distribute and save transaction data. All data on interpersonal (P2P) transactions are recorded in a ledger that is saved in blocks. These blocks are linked with each other in the form of a chain, generating a blockchain. After mining, blocks have a sequentially connected chain structure along with time stamping, which ensures liveness and probabilistic safety [1].

However, blockchain in its current form is too slow to be used in real services [2,3]. Because blockchain features timestamp-based recordings, it is very difficult to manipulate data. For this reason, blockchain has been applied to cryptocurrency, databases, artificial intelligence learning in big data storage, application security services, finance, gaming, social networking services, and many other areas. For the stable application of blockchain, it is necessary to improve the performance of blockchain networks and address the issue of scalability regarding the blockchain trilemma [2,3]. It is impossible to address all three issues of the blockchain trilemma: decentralization, scalability, and security. This concept is similar to that of the consistency–availability–partition tolerance theorem, also known

as Brewer's theorem [4], in the traditional field of distributed systems. Although it is difficult to address these three challenges, trials have been gradually conducted to ideally address the blockchain trilemma [3,5,6]. Public blockchain networks have the advantage of decentralization, whereas private blockchain networks [7] have the advantage of scalability. The nodes participating in the consensus method at the time of block mining and the triangular points of the blockchain trilemma are changed depending on the network structure, and this is where each point's trade-off occurs.

The problem of low scalability in a public blockchain arises when a transaction is recorded in a block or when nodes reach a consensus. A state-of-the-art method must be applied to improve the blockchain ecosystem, especially for proof-of-work (PoW) public networks. To solve the low scalability problem of PoW consensus, many trials for transaction per second (TPS) improvement, such as H/W improvement, Segwit, Hardfork, Sharding, and consensus algorithm changes, have been conducted. For example, Bitcoin Cash, a type of Hardfork that increases TPS by enlarging the block size, is a new blockchain different from conventional Bitcoin [1]; therefore, it is difficult to say that the method fundamentally addresses the scalability issue. Because a blockchain is an unstructured overlay network, propagated nodes can be propagated again; this is an inherent characteristic of blockchain. Therefore, it is difficult to improve the scalability. However, propagation and verification are achieved quickly through the Internet and H/W improvements, an innovative performance improvement in H/W is far from realization. In addition, it is impossible to ignore the power consumption generated by mining. Each improvement method has its own limitations. To fundamentally improve public blockchain scalability, it is necessary to study consensus algorithms.

During consensus after block mining in the conventional PoW blockchain Mainnet, processing block propagation takes the longest. We observed the block data log on the block explorer and found that block propagation is relatively slower than other processes, including block interval and block verification. Therefore, we resolve scalability problems by shortening block interval and propagation, vice versa. As such, it is important to study the block propagation process and block intervals. A block interval refers to the time it takes for a node to solve a nonce, which is defined as difficulty. The block interval of Bitcoin is 10 min on average [8], while that of Ethereum is 15 s [2]. In terms of the block propagation process for propagating 100% of the block created to all nodes, Bitcoin and Ethereum require 12.6 s on average [9]. The block verification process refers to the process of verifying a block in each node. More specifically, in the block verification process, after a header is hashed completely, a node checks if there is a number with a difficulty level, eventually accepts block propagation, and verifies that the propagated block has the hash of its previous block. Generally, it takes 10–12 s to finalize the block propagation process.

To address the scalability issue, it is important to further reduce the current difficulty and block interval. Even if the interval of mining a new block is shortened through reduced difficulty and changing the total hash rate, it is necessary to maintain the same security level as in a conventional Mainnet to solve the blockchain trilemma. Therefore, in this study, we investigated the relationship between block interval and block propagation rate and how to increase scalability in security maintenance. If the block interval time data continue to accumulate in the log, it is possible to derive an ideal consensus time using machine learning. In the experiment conducted in this study, the scalability issue of public blockchain was somewhat addressed. Accordingly, it was possible to lower the constant rise in difficulty and reduce H/W resources, which are the disadvantages of PoW consensus.

The remainder of this paper is organized as follows. In Section 2, related work is described. In Section 3, the experimental methodology is described to show that shortening the block interval is effective in reducing consensus time. In Section 4, we present the results for our established hypothesis, and finally, Section 5 concludes the paper.

2. Related Work

2.1. Background

Although the greatest advantage of the PoW blockchain is its strong security, an unnecessary increase in the difficulty level of PoW is generated. With an increase in the mining difficulty level, individual miners fail to receive mining rewards. Equipment for executing operations requires high specifications, and thus, excessive energy consumption occurs.

The scalability issue with blockchain arises when a new transaction is completed and nodes propagate a block, including the new transaction data, to their nearby nodes. Each node verifies that the received transaction is reliable. For this reason, with an increase in the number of nodes, it takes a longer time to deliver a new transaction to all nodes and to verify transactions. Consequently, the transaction speed gradually decreases. To address the scalability issue, public blockchains (Bitcoin and Ethereum), which employ the PoW consensus algorithm, must improve the speed of transaction consensus between nodes.

A stale block is a block that fails to be linked with a main chain and exits separately when two blocks are approved at the same time. If the block generation speed increases, the probability of generating stale blocks is higher, and security becomes weaker.

2.2. Scalability Issue in Public Blockchains

To improve the efficiency of the platform to which a blockchain network is applied, it is necessary to improve the scalability of the PoW consensus-based public blockchain [7,10]. Decentralized applications (dApps), which are developed with the application of the smart contract of Ethereum, are less universal owing to their low execution speeds and high gas fees. The problem of a high gas fee is almost solved with the use of the InterPlanetary File System protocol. However, the low execution speed remains. Although PoW is evaluated to solve the problem of Byzantine failure almost perfectly in the field of distributed network computing, it takes a long time to perform a consensus process. Therefore, its TPS is relatively low, as shown for Bitcoin (7 TPS or lower) and Ethereum (30 TPS or lower). Because it is difficult to solve the problem of low TPS in PoW consensus, there have been trials of delegated proof-of-stake (DPoS) based on proof-of-stake (PoS). Furthermore, efforts towards the decentralization of RAFT [11], Byzantine fault tolerance (BFT) [12], practical BFT [13], and centralization in private blockchains have ceased.

In this study, many different trials to improve scalability in PoW consensus research were conducted. The purpose of this study was to improve the efficiency of public blockchains that everyone can join. PoW guarantees a decentralized environment relatively and thoroughly. A comparison of consensus algorithms besides PoW is performed analogously to [14] with no additional considerations.

2.3. Previous Studies on Solutions to Public Blockchain Scalability

Table 1 lists the methods used to address the scalability issue in existing public blockchains, such as Bitcoin and Ethereum. As solutions to scalability issues, multiple methods, including on-chain, off-chain, child-chain, and inter-chain methods, have been suggested. Each method has its own advantages and disadvantages, and each has actual application cases [3,15,16]. For on-chain methods, there are big block, segregated witness (Segwit), and sharing methods. Big block is a method of sizing up a block. The Segwit method is a method of separating and saving a digital signature (which causes transaction speed to decrease owing to a limited block size in the blockchain). The sharing method is a method of partitioning and saving the processed block of the main chain and inspecting the validity of an allocated block to send it to the main chain. The big block method has been applied to Bitcoin and has been hardforked to Bitcoin Cash. It increases the generation rate of orphan blocks and causes the possibility of mining pool occupation for the entire hash rate. The Segwit method has the problem of fungibility occurrence, and the Sharding method is at risk from a 1% attack [17].

Table 1. Comparison between Existing Scalability Solutions.

Features	On-Chain		Off-Chain		
	Big Block	**Segwit**	**Lightning Network**	**Raiden Network**	**Sharding**
Solution	Scale up block size	Save digital signature separately	Record off-chain Tx value	Record off-chain Tx value	Partition and save block
Problem	Generation of orphan block	Fungibility occurrence	Lack of project verification	Lack of project verification	Integrity issue
Blockchain Project	Bitcoin Cash	Bitcoin	Bitcoin, Litecoin	Ethereum	Zilliqa

As off-chain methods, lightning networks and Raiden networks are simple solutions in environments other than blockchain, i.e., on-chain. The history of transactions recorded off-chain is not completely reliable. In addition, child-chain and inter-chain solutions have been proposed. However, they are different from the fundamental on-chain solutions proposed in this paper. One study [5] considered the last one-decade transaction data increase of Bitcoin and Ethereum and raised the scalability issue with the PoW algorithm. To address the scalability issue, it proposed the Bitcoin Lightning Network Protocol and a new DPoS consensus applied to EOS. Off-chain or on-chain scaling is problematic. The new method proposed in this paper simply compensates for the limitations of the application of the Sharding method.

Typical public blockchains attempt to address the efficiency issue of the consensus algorithm for the first first-generation PoW blockchain. For example, Silvio Micali's Algorand [18] aims to solve the trilemma. He suggested that it is possible to solve the blockchain trilemma through pure PoS (PPoS) consensus [19]. However, in the Algorand developed on the assumption that most participants are honest while some are not, there is no incentive for work (mining, delegating, staking). In this aspect, Vitalik Buterin suggested that it is difficult to make continuance without an incentive. There is an opinion that the consensus method without an incentive in Algorand fails to overcome the limitation that a network is unable to run correctly in a public blockchain [20]. The third generation blockchain EOS network undergoes a change in the consensus algorithm employing DPoS. Therefore, this is not a performance improvement case in PoW. This study focused on improving scalability in the PoW of a public blockchain. (See Table 2).

Table 2. Comparison between Our Proposed Method and the Others.

	Scalability Improvement	**Consensus Modification**	**Block Manipulation**	**Fundamental Solution**	**Trilemma Solution**
On-Chain	√	√	×	×	×
Off-Chain	√	×	√	×	×
Our Method	√	×	×	√	√

2.4. Public Blockchain Scalability Solution and Security Issue

A trial to address the scalability issue in the public Bitcoin blockchain is found in the method for problem-solving in [8]. With a change in the block interval, the block propagation time and stale block rate were compared. Finally, throughput was considered as a TPS improvement. A significant result was obtained in this study. It was found that an average of 15 s was not the optimal block interval.

To further solve problems with scalability, it is possible to reduce the consensus processes and shorten the consensus time. However, the following typical security problems arise:

- Selfish Mining Attacks. Selfish mining attacks occur when an attacker withholds a successfully mined block without propagation over the network and attempts to propagate the withheld block over the network if the network occupation rate of other

nodes exceeds that of the attacker. In this way, an attacker tries to maintain a high occupation rate and monopolize the mining rewards.

- Double-Spending Attacks. Double-spending means that a currency can be spent twice. Double-spending attacks use the interval of time that it takes for the blockchain consensus algorithm to process and verify each transaction. These attacks can incapacitate the consensus algorithm by occupying 51% of the P2P network nodes.
- Eclipse Attacks. Eclipse attacks occur when multiple nodes communicating with a particular node have a malicious attempt to attack the node.

We considered, in the case of common attacks, what can be possibly found in public blockchains referred to [21]. These attacks are caused by the consensus mechanism and block verification phase. Thus, in this study, we carefully keep in mind these common specific vulnerabilities. To respond to selfish mining attacks, the test environment of this study considered the mining power and block propagation rate. To lower the probability of a 51% attack, a scenario of mining competition between mining pools was added to the test environment. It was designed to maintain the majority of the mining pools and the majority of the adversary rate. In the design for solving the problem of eclipse attacks, virtual nodes were diversified and randomly selected.

3. Adjusting the Block Interval in PoW Consensus

To find an optimal block interval in the PoW consensus algorithm proposed in this study, we conducted a four-step experiment (see Figure 1). The purpose of this experiment was to verify that, along with the increase in the average hash power of nodes participating in Ethereum, even if the difficulty level with the right upward tendency is somewhat decreased, there is no significant issue with Mainnet maintenance. Moreover, we suggest reference information when a consensus process is designed in a PoW consensus-based blockchain.

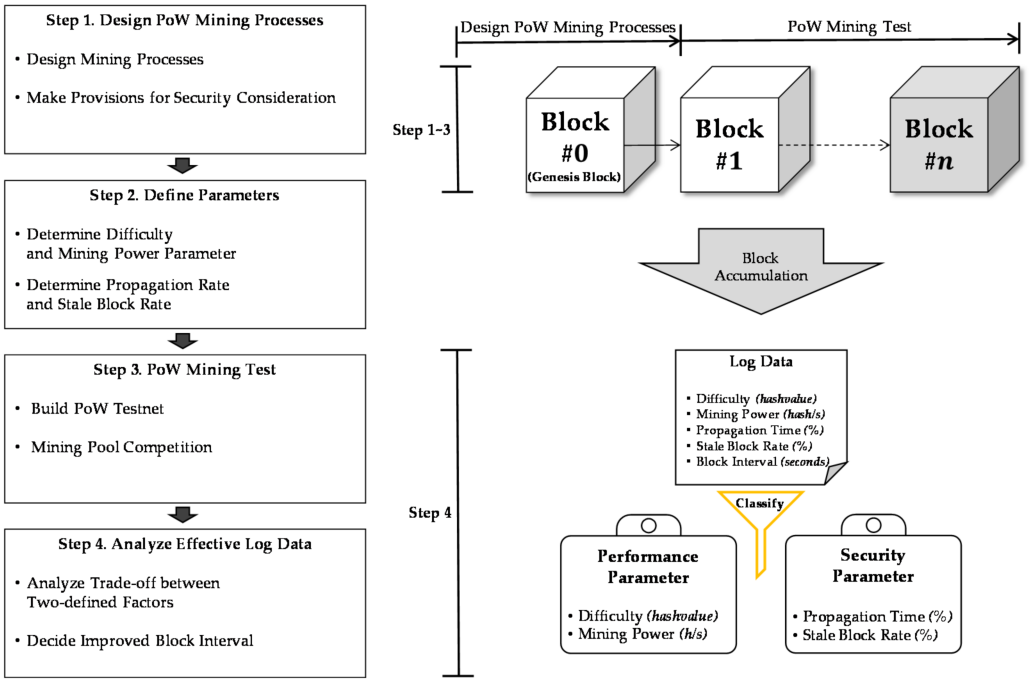


Figure 1. Process Overview of Adjusting Block Interval in PoW Consensus.

The first step was to establish a Testnet (or test network). In the experiment, the comparison indexes were the block interval time and block propagation rate. In the Testnet, these index values should be extracted after blocks are accumulated for a certain period. In this step, an appropriate block time was set, and a new generation of blocks was designed. If the value of difficulty is low, the entire blockchain network has weak

security. Therefore, three types of typical blockchain attacks and proactive approaches for the attacks were considered to prevent the security of the entire blockchain network from being compromised. The second step was to define the parameters for determining two indexes that are necessary in the first step. The third step was to perform mining simulations and accumulate miner and network log data. The fourth step was to analyze the valid log data and derive the experimental results. In the last step, the PoW block interval in an Ethereum test environment is suggested.

3.1. Designing PoW Mining Processes

In the first step of PoW modeling, PoW mining is divided into three parts: block interval, block propagation, and block verification. In this step, a network is established that is as similar to a real Mainnet as possible. This is because accurate results depend on the similarity of the network to a real Mainnet. The test environment of the proposed method was aimed at improving the TPS of the Ethereum network, which is widely applied to various areas, including cryptocurrency and dApp design.

3.1.1. Designing Mining Processes

The experiment of this study aimed to establish a PoW consensus Testnet that is as similar to a real network as possible. To achieve a consensus process to link blocks in the form of a chain, it is necessary to mine the blocks. Block mining consists of three processes: block interval, propagation, and verification.

The algorithm for Ethereum mining is illustrated in Figure 2. In the algorithm, the blockchain to test is designed in a directed acyclic graph. It is made up of an infinite number of blocks owing to its infinite directed acyclic graph. A link is created in the direction from a block to a different mined block. The general PoW processes are presented on Line 5. The consensus function is shown on Lines 6 to 12. In these lines, the consensus process of Ethereum is presented as an example of an experiment. When the consensus process of Ethereum, somewhat different from other consensus processes of a PoW blockchain, is designed, a miner is highly likely to suggest a new block without any propagation of the latest mined block while TPS is being improved. For this reason, a temporary fork is preferred. Therefore, to prevent any unnecessary effort for mining during a temporary fork period, a Greedy Heaviest Observed Subtree (GHOST) consensus algorithm was designed. Unlike the Bitcoin consensus, the GHOST consensus protocol repeatedly selects the next block of the root of the subtree with the largest number of nodes (see Figure 2). As shown on Lines 14 to 20, the current codes of Ethereum select a sub-branch according to the difficulty level of the crypto(nonce) solved without any comparison of the subtree size.

In the block interval process of Ethereum mining, it is possible to measure the block interval time. By setting the Ethereum block interval time in the PoW consensus algorithm to be as fast as possible, it is possible to address the scalability issue. In the block propagation part, the block propagation rate was measured. Therefore, when block propagation was performed at an appropriate level or higher in the fast block interval time, it was possible to check whether the entire consensus process has a problem.

After the Testnet reflecting a difficulty level lower than the actual one was established, the lower bound of the block interval influenced the hash rate of the entire blockchain. For this reason, the occupation rates of the mining pool and adversary mining pool were changed. With a reduction in the difficulty level, the hash power of the adversary (competing) mining pool was relatively increased, and thus the occupation rate of the mining pool can fall to 51%. The point at which the occupation rate of the mining pool is 51% does not indicate an ideal block interval for significant TPS improvement. If so, the chance that a competing chain takes over the original chain, e.g., by a selfish mining attack, is greatly increased. The issue that arises when a block interval is shortened is that it is difficult to synchronize it perfectly with the nodes placed in the world. No matter how fast the Internet is, it takes some time to exchange information between nodes over the Internet. This is called latency. Although it is not too long, it is enough time to cause a collision

in which transactions have an inconsistent balance history. A node continues to mine a block over the previous block until it hears that a new block is discovered, and it throws its mining block away when the new block appears. After a valid block is discovered in a different position in the network, an uncle block continues to be mined. This is called a stale block. As the block interval is shortened, the probability of stable block occurrence increases. The higher the number of stable blocks, the more vulnerable the network is to attacks. Because the point at which the occupation rate of the mining pool is 51% is not ideal, this study conducted a comparison analysis of block propagation rate and aimed to find an improved block interval by conducting a comparison analysis of the block while keeping the stale block rate at a certain level. With an increase in the stale block rate, it is necessary to improve the efficiency of the mining pool and prevent the possibility of monopolizing an incentive.

Adjusting Block Interval in PoW Consensus Algorithm		**Description**
1:	$L = < B, P >$, *graph of blocks B, pointers P, local Blockchain at node x is directed acyclic*	**General PoW Processes**
2:	*b, a block record with fields:*	
3:	*parent, the block preceding b in the chain*	
4:	*pow, the proof − of − work nonce of b that solves the crypto(nonce)*	
5:	*children, the successor block of b in the chain*	
6:	*propose*():	**Consensus Function**
7:	*while true do*	
8:	*nounce = local − random − coin*()	*- perform local check to obtain a nonce*
9:	*create block b : b.parent = last − block and b.pow = nonce*	*- create a new block*
10:	*if solve − crypto (nonce) then*	*- if the chosen nonce solves the nonce*
11:	*broadcast*(({*b*}, {(*b*, *b.parent*)}))	*- broadcast to all*
12:	*decide*(*b*)	*- consensus reached*
14:	*get_main branch*():	**Select the Branch with the Majoraty Nodes (Greedy Heaviest Tree)**
15:	*b* ← *genesis block*	*- start from the blockchain root*
17:	*B* ← *B* ∪ {*block*}	*- update vertices of main branch*
18:	*P* ← *P* ∪ {(*block*, *b*)}	*- update edges of main branch*
19:	*b* ← *block*	*- move to next block*
20:	*return* ⟨*B*, *P*⟩.	*- return to the Ethereum main branch*

Figure 2. Adjusting Block Interval in PoW Consensus Algorithm.

3.1.2. Provisions for Security Consideration

To implement a Testnet, it is important to make the network similar to a Mainnet. Nowadays, a Mainnet is joined by global nodes so that it is relatively safe from the three types of attacks described in Section 2.3 (selfish mining attacks, double-spending attacks, and eclipse attacks). A Testnet consists of virtual nodes. Therefore, its environment was established to be as similar to that of the real environment as possible. For the three types of typical attacks for a PoW blockchain network, proactive approaches were devised as follows:

- Proactive Approach for Selfish Mining Attacks.

The experimental environment in this study considered the mining power and block propagation rate for responding to selfish mining attacks. When a miner or mining pool with strong hash power propagates a generated block intentionally late, a blockchain instantaneously encounters a fork and selects the branch (see Figure 3).

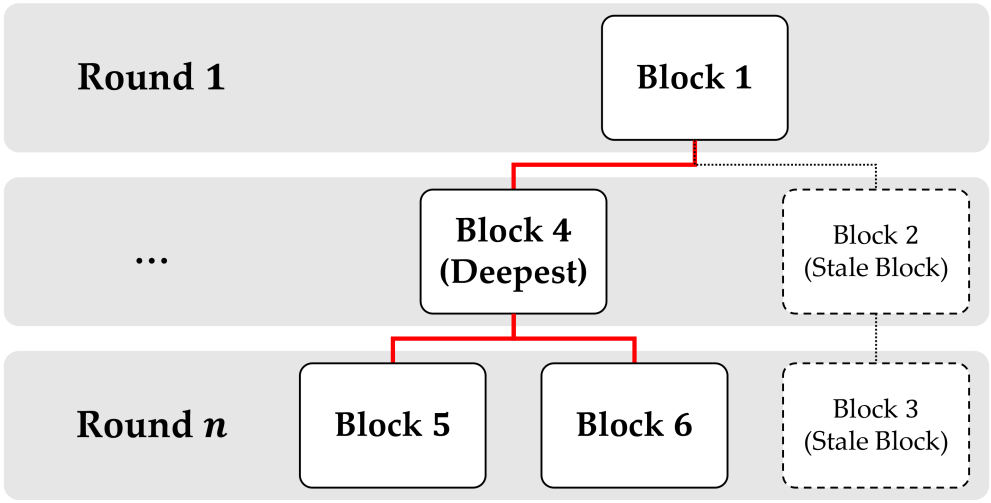


Figure 3. Proactive Approach for Selfish Mining Attacks.

- Proactive Approach for Double-Spending Attacks.

The proactive approach for double-spending attacks is to defend against the most widely known and strongest attack, a 51% attack. When hackers control 51% or more of a network's mining power, it is possible to perform three kinds of attacks (see Figure 4). First, while controlling 51%, the hacker is capable of reversing transactions and performing double-spending. Second, the hacker can possibly make a particular transaction, or all transactions fail to be accepted. Third, the hacker can prevent a particular miner or all miners from mining a confirmation block. Although a miner has 51% or more of a network's mining power, there are things that the miner is unable to do. To defend against this attack, network members should establish full nodes. Light nodes are vulnerable to attacks. This is because light nodes trust a particular miner or a full node.

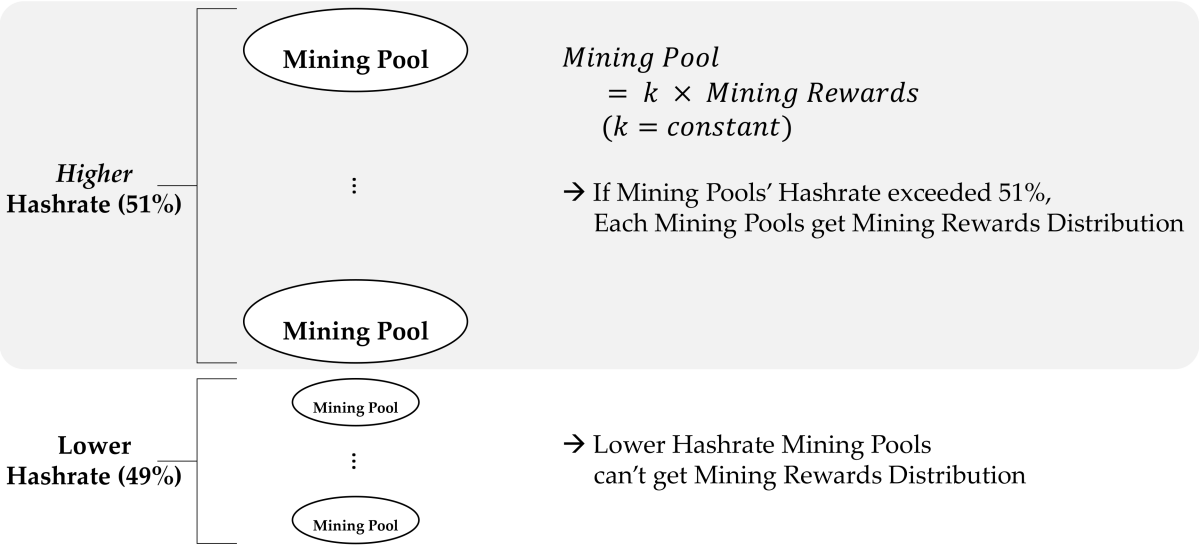
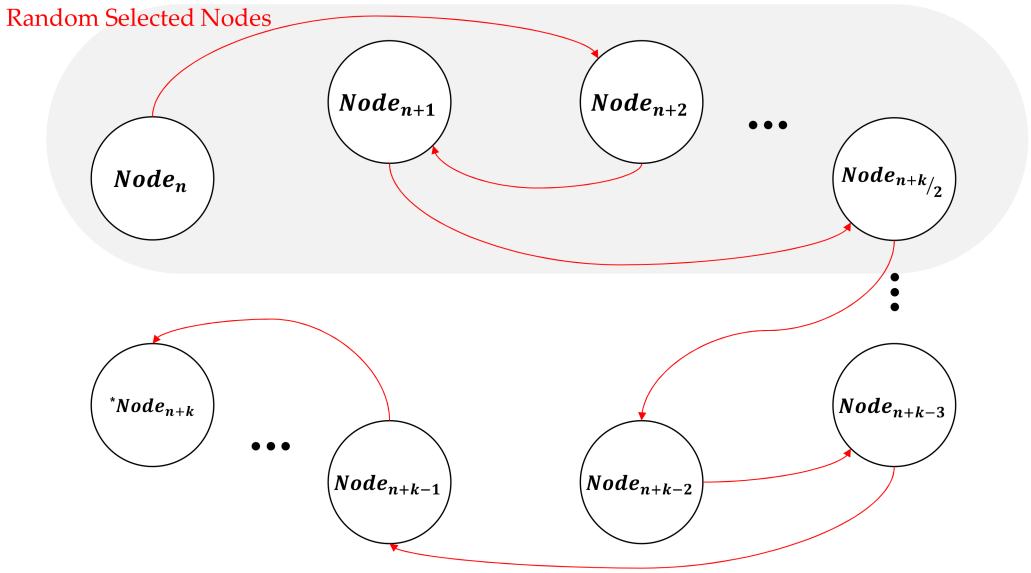


Figure 4. Proactive Approach for Double-Spending Attacks.

- Proactive Approach for Eclipse Attacks.

The proactive approach for eclipse attacks in Ethereum is to select a node randomly. For node selection, if a node is randomly selected among $n + k$ distributed peers in the entire network, an attacker can guess a node that is necessary for designating a target. Another approach is to limit the number of nodes per IP address or computer, thereby

saving a different node's information in information storage whenever it encounters a different node. If a node leaves the network and joins it again, and then continues to access the information, it is possible to connect some legal peers before a different node is found. With an increase in the number of links, the network speed decreases. If more links, though not infinite, are allowed, it is more likely to obtain a node linked to a user (see Figure 5).



* Number of nodes are dependent on all nodes in the blockchain network.

Figure 5. Proactive Approach for Eclipse Attacks.

3.2. Defining Parameters

The next step after designing PoW mining processes is to define the scalability and security parameters that are used when the improved block interval time of the PoW model is determined.

Figure 6 illustrates the parameters to be measured in the experiment with two groups: performance and security groups. Difficulty and mining power based on PoW consensus is defined as the performance parameters. To ensure proactivity for blockchain security issues, the propagation rate and stale block rate are defined as security parameters. When the PoW consensus difficulty is changed, the propagation rate is extracted as a security parameter. Finally, an ideal adversarial mining strategy, including security provisions, is determined.

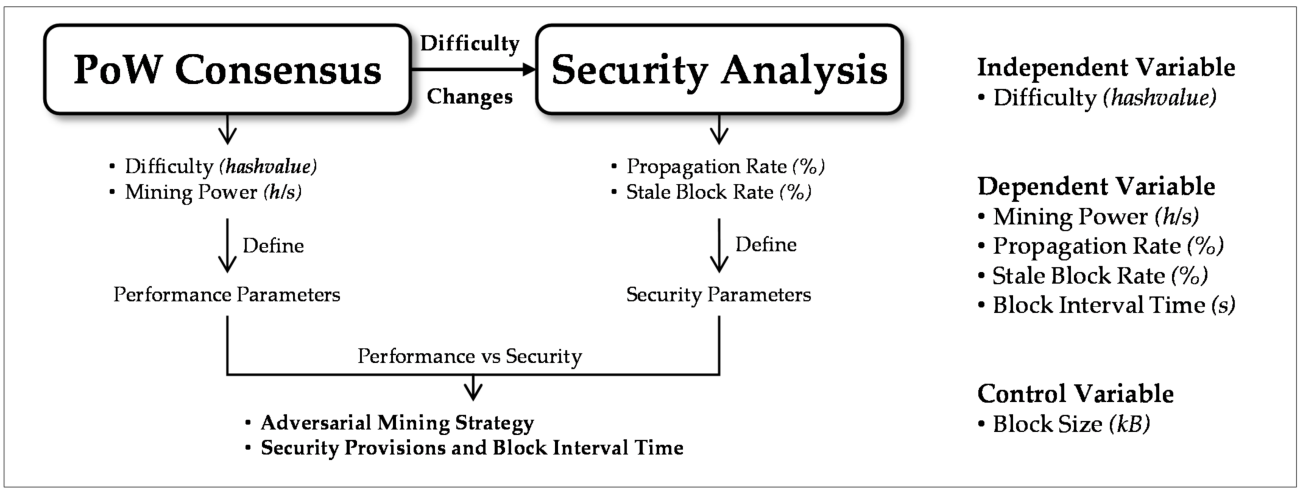


Figure 6. Define Parameters into Two Groups.

Four parameters in this study were defined as follows:

- Difficulty. By adjusting the difficulty level, it is possible to influence the block interval time. Difficulty was used as a performance parameter, and logs were collected at the time of mining competition.
- Mining power. One of the measurable data in a Testnet is the mining power of a mining node. It is possible to measure the mining power of the mining pool in the Testnet. The mining pool with the highest importance, or the mining pool with 51% or more of the mining power, will achieve mining success and will receive Ethereum rewards. On balance, this is a parameter in the network similar to the real Ethereum Mainnet, where the competition ratio of major pools and other adversary minor pools remains at 51:49. According to the mining power, it is possible to prevent selfish mining attacks. Therefore, mining power was used as the first security parameter of the Testnet.
- Propagation rate. The results of this study were influenced by the effective throughput of the network. The effective throughput is the rate of nodes that propagate a block in an average block interval. In other words, the rate of nodes that receive and verify the latest block over the entire network in an average block interval is 83%. If the effective throughput is 90%, it means that among the nodes of the network with a relevant bandwidth, 10% have a delay. Denial of service can potentially occur, weakening the effective mining power of the network. In the block propagation process for propagating a successfully mined block to the entire blockchain network, the propagation rate should reach as close to 100%. Therefore, to defend against double-spending attacks, the propagation rate was used as the third security parameter.
- Stale block rate. In terms of Ethereum mining, as in Bitcoin mining, it takes a long time to propagate a block using a high gas price. It also takes a long time to verify a change in the state transition. It is necessary to maintain the stale block rate and shorten the block interval. In the block mining process, the rate of stale blocks is inversely proportional to the security of the entire network. Therefore, the stale block rate was used as the second security parameter.

3.3. PoW Block Mining Test

The three-step PoW block mining test performed a mining simulation for extracting log data based on the parameters defined in Section 3.2.

In Figure 7, blocks from the genesis block with block height '0' to the block of height '*i*' are the decided blocks. Blocks with block heights from $i + 1$ to $i + k$ are the ones to mine (undecided blocks) and will be linked next to the block with block height i. The process of Testnet mining is illustrated in Figure 1. Mining was performed in the Testnet. After the undecided block with block height $i + 1$, mining was performed at the time interval of the previous block. Therefore, in the previous mining step, the level of difficulty was adjusted.

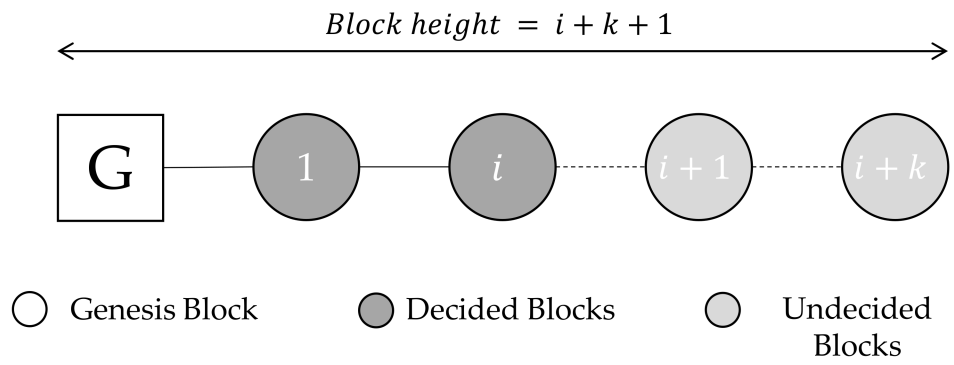


Figure 7. Testnet Mining Processes.

Figure 8 illustrates the method for determining a subtree using the five parameters defined in Section 3.2. To shorten the block interval along with a change in the level of

difficulty, it is necessary to ensure the presence of stable blocks, as shown in Figure 8. When a block is propagated, it is necessary to check the block propagation rate and maintain the final stale block rate. Bitcoin and Ethereum, as typical PoW blockchains, have different consensus policies for accepting stale block rewards. The situation for each step in Figure 8 is described as follows:

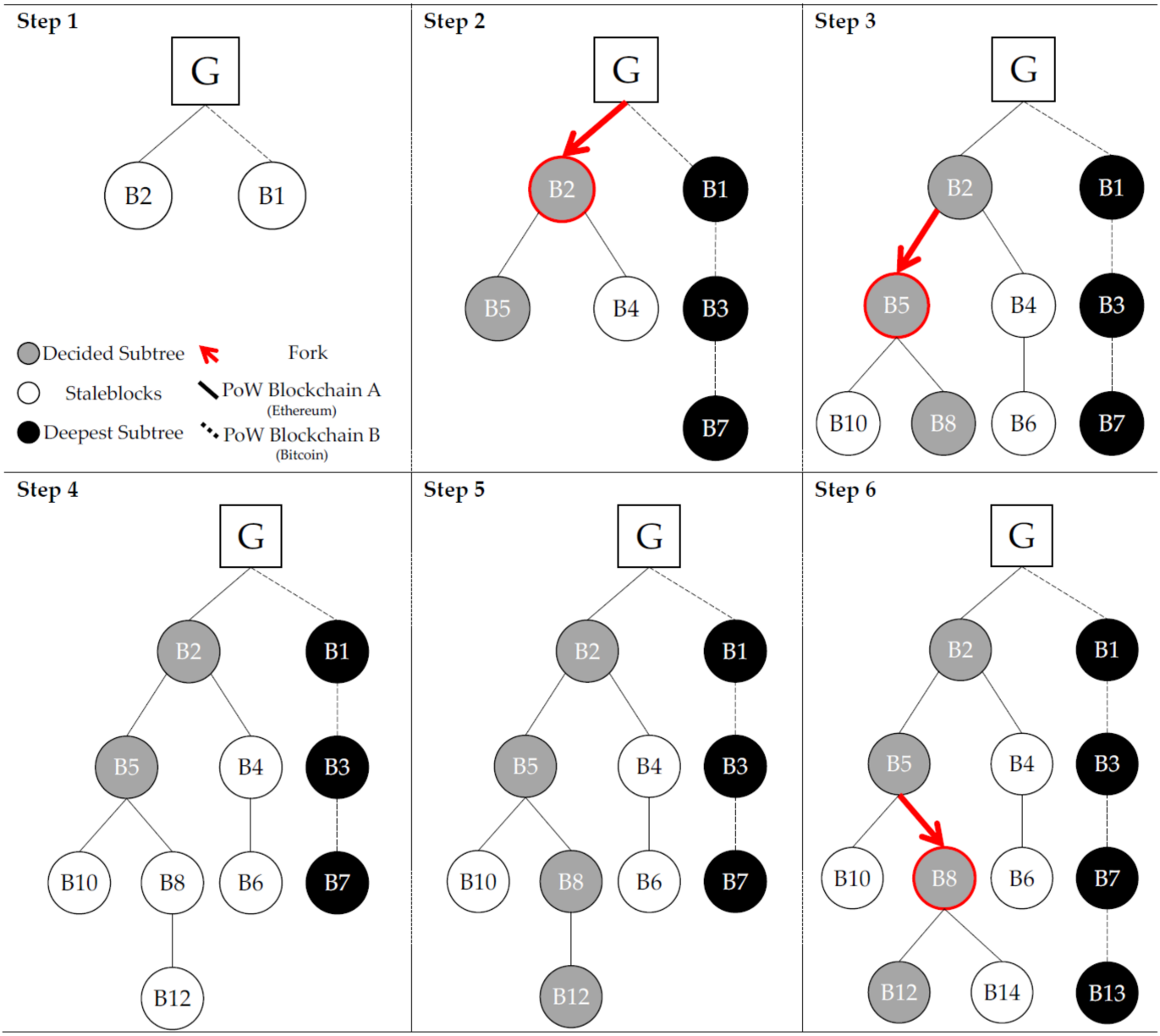


Figure 8. Subtree Deciding Process between Two Different PoW Blockchains (e.g., Bitcoin and Ethereum).

- Step 1: The genesis block and blocks B1 and B2 are mined. A fork occurs in the main chain. In this step, PoW blockchains A and B represent Ethereum and Bitcoin, respectively.
- Step 2: In terms of the subtree of B1, the longest tree is selected, and blockchain B is generated. In the case of blockchain A, B2 was selected, and forking was completed. The subtree B2 was relatively heavy.
- Step 3: G-B1-B3-B7 was generated in blockchain B. In the B2 subtree, forks B4 and B5 occurred again. Because the subtree of B5 is heavier than that of B4, B5 was selected, and forking was completed. B4, as the uncle block, received the mining reward.
- Step 4: Below B8 as the final block of blockchain A, block B12 was linked.
- Step 5: In blockchain A, G-B2-B5-B8-B12 was generated.

- Step 6: In blockchain B, block B13 is normally linked to the B7 sub-block. In blockchain A, the subtree of B8 is the heaviest. Therefore, B8 was selected, and the forking was completed. The G-B2-B5-B8-B12 blockchain was generated. The new B12 and B14 block fork arose. The dotted line (black) in Step 6 of Figure 8 represents the mining reward consensus process for Bitcoin.

In Step 1, from the initial genesis block, a fork into blocks B1 and B2 occurs. After the process, the fork type of Bitcoin is different from that of Ethereum. Step 2 shows that the mined blocks are divided into the heaviest and deepest trees. After the process, in Steps 3 to 6, when blocks are found at the time of forking, Bitcoin selects the deepest subtree and discards a short blockchain. If a block interval is shortened in Bitcoin, more blocks are generated in a shorter time, and information created by more and newer blocks should be propagated throughout the network. Therefore, more stale blocks are generated. For this reason, it is necessary to set a sufficient block time during which information processed in the Bitcoin network is delivered to all nodes.

According to the decided subtree in Steps 2 to 6, the Ethereum network does not select the deepest subtree between B1 and B11; instead, it selects the heaviest subtree (gray). Given the characteristics of the GHOST consensus protocol, the root of the subtree with the largest number of nodes (heaviest) is selected as the next block. The Testnet mines a block of the subtree based on the difficulty of solving the crypto(nonce). When this process is repeated, the process of generating the heaviest subtree is not influenced by the stale block rate, and the block interval is shortened. In a series of processes, the difficulty level is adjusted, the block interval bound is lower, and the scalability issue is addressed. Nevertheless, a lower level of difficulty and shorter interval of mining a new block leads to a reduction in the block propagation rate and an increase in the stale block rate. Consequently, security issues arise. The block generation speed, security, and stale block rate are all related. Bitcoin, with a low block generation speed, has a low, stable block rate and high-security level. Ethereum, with a fast block generation speed, has a high stale block rate and low-security level. To solve the problem of centralization, Ethereum rewards the orphan blocks. Blockchains with a fast block generation speed face weak security owing to their high stale block rate. This is because block propagation requires a certain amount of time over the network.

4. Experiment and Results

In the three-step-based experimental environment (designing the PoW model, defining parameters, and simulating the mining) described in Chapter 3, mining consensus simulation was performed in the range of block interval shortened from 25 to 0.5 s. Throughout the experiment, the security and performance were analyzed. The objective of this experiment was to observe the transactions at the checkpoints applied for Ethereum by changing Geth, thereby finding a block commit delay in the blockchain. In Section 4.3, the final block interval time is determined under the conditions of security (environment) and performance (consensus).

4.1. Building the Experimental Environment

As shown in Figure 2, after adjusting the block interval in the PoW consensus algorithm, a consensus algorithm was first established. Then a test was conducted. A specific mining power was given to each mining pool, and the PoW for a miner was simulated. In a block interval, a new block belongs to a miner. As shown in Figure 8, it was assumed that in a conventional PoW blockchain, a miner mines the first block received, and a fork is solved in accordance with the rule of the heaviest chain. When a fork is solved, the block that fails to contribute to the main chain is a stable block. In the simulation, different blocks' changes in difficulty level were not considered. Therefore, the longest chain was determined according to the number of blocks. For node links, a channel of points between nodes was generated, routers and switches were abstracted, and wait time and bandwidth

were considered. Because this study was also focused on the influence of difficulty level on block interval, transaction propagation was considered implicitly.

In the experimental environment, the configuration of the Ethereum Testnet, which was as similar to the real one as possible, as emphasized in Section 3.1, was validated. Table 3 lists the five validation factors of the experimental environment of the Testnet, in accordance with Table 2. The experimental environment was established to simulate the PoW consensus algorithm for a miner, to analyze mining competition between many different mining pools, and to find a block interval.

Table 3. Testnet Configuration.

Category	Description
OS	Ubuntu v20.04.01
Virtual nodes	Docker Compose v1.27 [22]
Environment	Geth v1.8.6 [23]
Language	GO language v1.9 [24]

4.1.1. Building Environment

In the experimental environment, the virtual nodes were based on Docker (see Table 3).

Each virtual node ran Geth (go-ethereum), the official client software offered by the Ethereum Foundation. When Geth runs at the beginning, it tries to connect to a different Ethereum client (node) in the network and downloads the genesis block of the local network. To keep a copy of the latest blockchain, Geth ceaselessly communicates with different nodes. Simultaneously, a mining node mines a block and adds a transaction to the blockchain, and a verification node verifies the block. The virtual nodes are based on Geth. In a real network, it is impossible to adjust a genesis block discretionally. For this reason, a blockchain local test network was employed. The Testnet was established to be as similar to the real Ethereum as possible with the design of virtual nodes. In the Testnet, the genesis block difficulty level was randomly set to be lower than the current difficulty level.

4.1.2. Environment Validation

To examine the hypothesis that a shortened block interval time does not harm the security, this study simulated the PoW consensus algorithm. When mining nodes joining PoW mining are designed, adversary nodes are created by the hash rate. After a new block was mined, a blockchain was established according to the PoW algorithm and the deepest subtree process. In the experiment, other variables, such as block size and network speed, were not considered; a change in hash power was analyzed to lower the difficulty level. As a result, it was found that although the hash power was low in the mining competition, there was no significant problem with security.

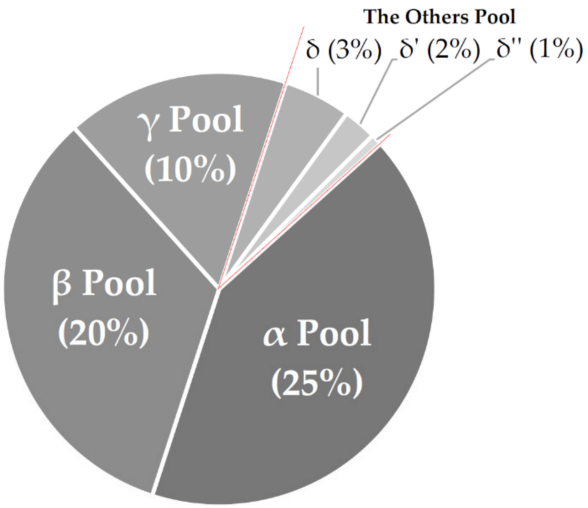
In a mining simulation environment, it is important to collect the block propagation rates of all nodes in real-time. When the block interval is shortened, a new block is generated after the previous block is successfully propagated to all nodes. For this reason, the block propagation rate in a Testnet should not be much lower than that in a conventional Mainnet. P2P was applied to node links. To verify that the Testnet (composed of routers, switches, etc.) was made as similar to a real Ethereum network as possible, validation factors, such as block size distribution and geometrical distribution of the nodes/mining pool, were defined, as listed in Table 4.

In this study, the block size was set to approximately 30 KB or more. In consideration of geographical locations (North America, Europe, Asia Pacific, etc.), network delay was given to simulate nodes joining the consensus process.

Table 4. PoW Blockchain Validation Factors.

Validation Factor	Description
Block Size Distribution	Variable Tx load
Geometrical Distribution of Nodes	Worldwide distribution
Distribution of Mining Pool	Worldwide distribution
Standard Mechanism	Default
Relay Network	Miner network

Figure 9 illustrates the distribution of mining pools in the Testnet. Nodes in mining competition were designed based on the hash power rate of the main mining pools in real Ethereum rather than virtually. The hash power distribution is important in the reward step of the PoW consensus process; therefore, it was designed in reference to the mining pool status of the real Ethereum network. In the distribution, the α mining pool with the most hash power accounts for approximately 25%, followed by the β mining pool (20%), γ mining pool (10%), δ mining pool (3%), δ′ mining pool (1.5%), and δ′′ mining pool (0.5%). A mining pool is defined as a mining cooperation voluntarily established by global mining companies to increase the mining success rate. Miners α to δ″ represent the current miner pools of Ethereum. Miners α and β are representative mining pools with multiple miners who compete for mining. The mining pools of Ethereum (top pools α, β, and γ) account for 55%. Therefore, it was assumed that they had harsh competition for mining rewards. The hash power distribution worldwide was set as shown in Figure 9. This is because it was necessary to solve problems with security in an assumable mining reward competition scenario.



Mining Pool	Description
α	Virtual mining pool with the highest hash power
β	Adversary mining pool with the second highest hash power
γ	Adversary mining pool with the third highest hash power
δ	Others (numerous virtual mining pools with lower hash power)
δ'	
δ''	

Figure 9. Hash Power Distribution in the World.

4.1.3. Data Collection Method

In Cases 1 and 2, after approximately 610,000 new blocks were mined, the mining process stopped, and the mined blocks were collected (see Table 5). This experiment was conducted twice to reduce the error range of the experimental results. The collected Ethereum transactions were observed to find a block commit delay.

By using the Docker virtual nodes designed in line with the distribution in Figure 9, we performed the experiment twice. In Cases 1 and 2, approximately 610,000 blocks were mined in the Testnet with an adjusted low level of difficulty.

Table 5. Overview of Experimental Cases Run.

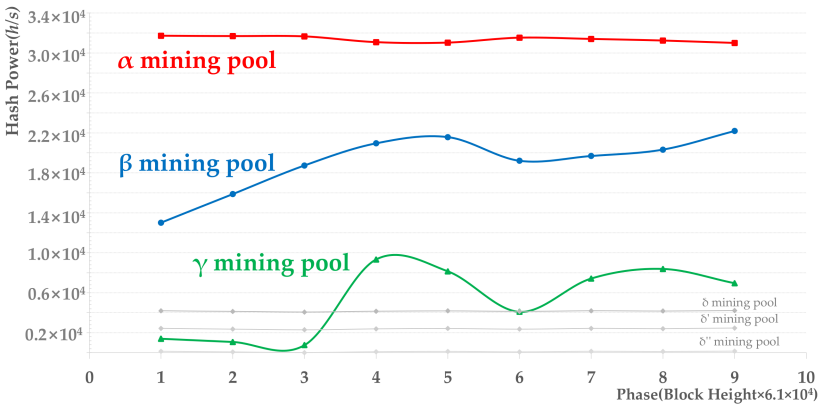
	Case 1	Case 2
Start of Tx Collection	2021-02-10 09:19 UTC	2021-02-16 13:11 UTC
End of Tx Collection	2021-02-11 08:25 UTC	2021-02-17 14:16 UTC
No. of Preceding Blocks	610,309	610,022

4.2. Evaluation Results

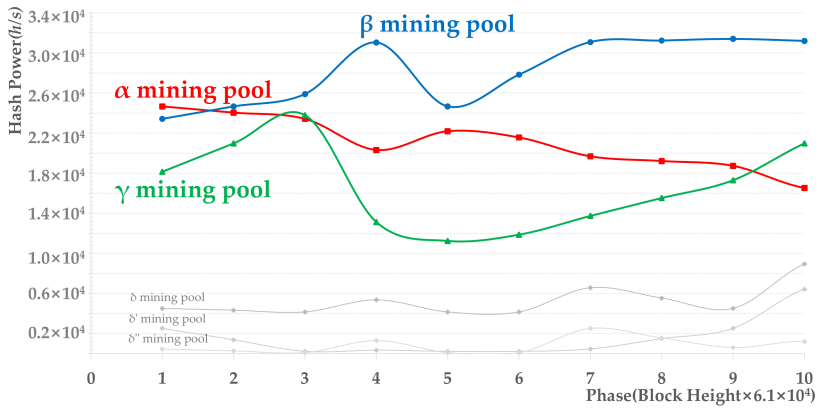
The adjusting block interval in the PoW consensus process performed in this study required two kinds of analysis. This is because it was necessary to improve performance along with the greatest reduction in PoW consensus time and to consider network security in the blockchain.

4.2.1. Performance Analysis

As shown in Figure 10, with a reduction in the level of difficulty, the hash power for block mining is also lowered. In Figure 8, the block height of 610,000 was divided by 61,000, and therefore, 10 phases were generated. To investigate the change in hash rate, the level of difficulty was adjusted, and the result of the mining competition was derived.



(**a**) Case 1



(**b**) Case 2

Figure 10. Mining Competition in Each Experimental Case: (**a**) Case 1; (**b**) Case 2.

In the first case, mining pool α maintained its hash power at a certain level. Mining pools β and γ, and mining pools that have lower hash power than mining pool δ (with low hash power), are in mining competition. As the level of difficulty was lowered, mining pool β with a 20% occupation rate and mining pool γ with a 10% occupation rate increased

their hash power more than in the beginning phase, while mining pool δ and others with low occupation rates did not.

In the second case, the hash power of α was reduced, and at the same time, β with the highest occupation rate and γ with the next highest occupation rate among adversary mining pools have mining competition. As the level of difficulty decreased, the hash power for block mining was reduced. Therefore, it was proven that the hash power of miner α was reduced, whereas the adversary miners β to δ′ tended to increase their hash power.

When the level of difficulty was reduced, the hash rate was crossed. With a reduction in the level of difficulty, there was no difference between miners α and δ in terms of hash power between α and β to δ″. Miner α with a 25% occupation rate and miner δ with a 3% occupation rate had a similar hash power, meaning that a small mining pool's hash power can influence the entire network. Ethereum, with a fast block time, has a high probability of stale block occurrence, which means its security is weak. Blockchains with a short block time face weak security owing to the high rate of stale blocks. If the level of difficulty is not sufficiently high, anyone can easily mine a block. In the block mining process, block generation and block propagation repeatedly occur within a short time, and thus the rate of stale blocks increases.

4.2.2. Security Analysis

There are two scenarios of stale block generation at the time of mining: (i) a stale block is generated when miners are connected to each other, and they try to gain the most profits even if the block generation process is not requested and (ii) when a block is propagated fast to most of the network. This is because whether the propagation of a previous node occurs is important. The influence of a block internal in a PoW-based blockchain on the median block propagation time and stale block rate was analyzed.

The simulation was performed in the block interval range from 25 to 0.5 s (see Table 6). At a block interval of 10 s, the propagation time was approximately 3 s, and the stale block rate was 1.82%. It was necessary to observe data in the block interval range from 25 to 10 s, which is similar to that in a real Mainnet. In the block interval range from 25 to 10 s, the propagation rate in Case 1 was 96%, and that in Case 2 was 91%. Therefore, the propagation time in Case 2 fell by approximately 5%. To check that the propagation time fell in a shortened block interval, a block interval of 9 s or shorter was considered. In Case 1, when a block interval was forced to be shortened from 9 to 7 s, the propagation time was shortened from approximately 2 to 1.43 s, falling by 39%, and the propagation rate dropped by 6%. In Case 2, the propagation time in the same time zone was shortened from 1.34 to 0.84 s, falling by approximately 59.5%, and the propagation rate dropped by 17%.

Table 6. Impact of Block Interval on Median Block Propagation Time and Stale Block Rate.

Block Interval (s)	Case 1			Case 2		
	Propagation Time (s)	Propagation Rate (%)	Stale Block Rate (%)	Propagation Time (s)	Propagation Rate (%)	Stale Block Rate (%)
25	35.73	96	1.72	25.66	96	0.16
15	14.7	96	1.51	10.65	96	0.13
10	4.18	96	1.82	2.91	91	0.16
9	2.08	95	2.15	1.34	88	0.35
7	1.43	81	2.54	0.84	71	0.45
5	1.21	77	3.20	0.67	69	0.86
3	1.00	60	4.77	0.35	61	1.73
2	0.89	52	8.64	0.37	56	2.94
1	0.84	29	16.65	0.40	28	6.98
0.5	0.82	31	26.74	0.53	17	12.44

According to the consensus time of the conventional Ethereum Mainnet, the block interval time was approximately 10 s or more. This is because it was necessary to secure a propagation rate of 91% to 96%. Given the simulation results, it is possible to determine whether the average block propagation time of Ethereum Mainnet is 15 s. Because Ethereum is an overlay network, it is aimed at propagating new block information to all nodes. In a P2P overlay network, nodes are connected to each other, share resources, and serve as servers and clients simultaneously. In such a network, nodes can share and exchange information directly with other nodes without the help of servers.

Table 7 lists the transaction throughput, block interval, and TPS calculated using Mainnet. In the block interval range from 9 to 7 s, TPS increased from 66.6 to 85.7 in the Mainnet. In the block interval range from 7 to 5 s, Case 1 showed an 18.1% reduction in the propagation time and a 4% reduction in the propagation rate. Case 2 showed a 25.4% reduction in the propagation time and a 2% reduction in the propagation rate. In terms of performance, TPS increased to 34.3 in the Mainnet.

Table 7. Block Interval and TPS Results in Testnet.

Tx in a Block (Tx)	Block Interval (s)	TPS (Tx/s)
600 (On Average)	**25**	24
	15	40
	10	60
	9	66.6
	7	85.7
	5	120
	3	200
	2	300
	1	600
	0.5	1200

On balance, if the block interval was shortened from 15 to 5 s in Cases 1 and 2, the TPS trebled from 40 to 120 s in the Mainnet. For this reason, in terms of performance improvement, the block interval range from 5 to 9 s was preferred over the block interval of 15 s as a consensus time. In the block interval range from 5 to 9 s, the propagation rate decreased to 23%, and the stale block rate was 1.21%. If the consensus time is shorter, the propagation rate would be 60–61% only when the block interval is shorter than 3 s. For this reason, a shorter consensus time is likely to cause security problems. When the block interval was shorter than 1 s, the propagation rate was 28–29%, and the stale block rate varied from 7% to 27%. In such a case, the decision tree process of the proposed algorithm determines that the heaviest subtree is selected incorrectly.

4.3. Performance vs. Security

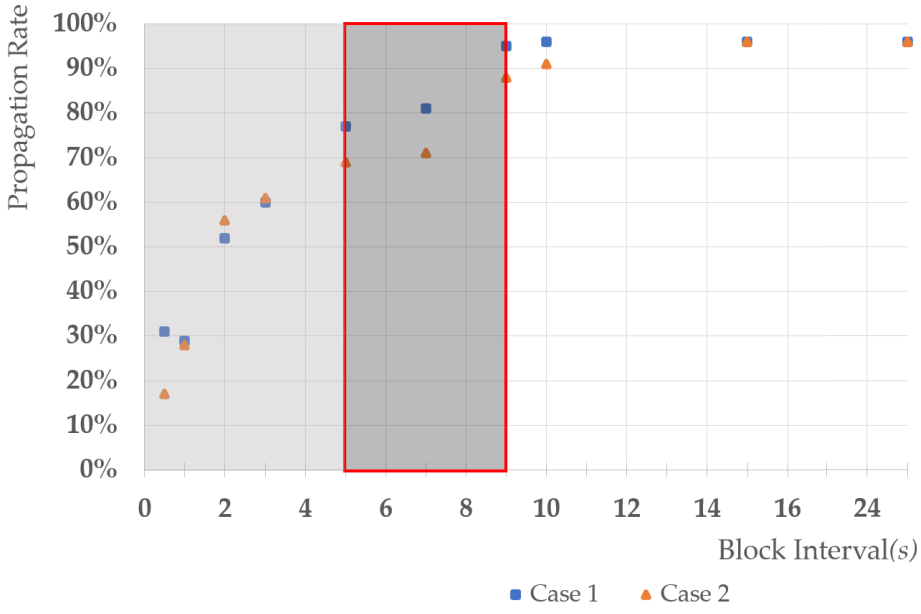
The network established as an experimental environment is an unstructured overlay network. Each node has no restrictions on selecting a neighboring node. To find a block propagation node, it propagates a message to the neighboring nodes sequentially. The network can have advantages, such as enclosure of transaction history and a flexible search for a live node, but this causes scalability problems. If the number of nodes is large, the network also has many messages and does not guarantee propagation to a target node.

In terms of performance and security in the experimental blockchain, the block size, block interval, transaction throughput, fork count, and security are correlated as follows.

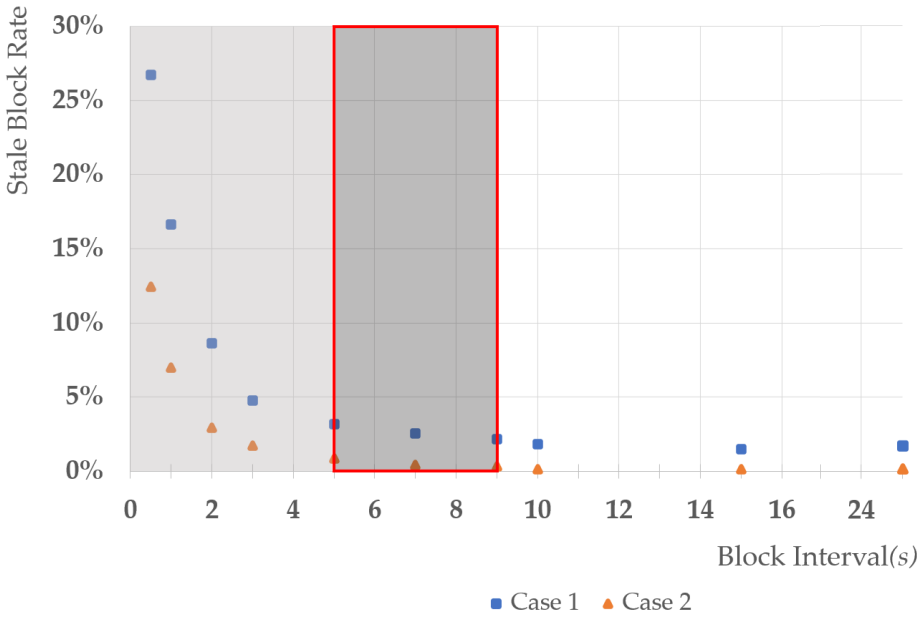
With an increase in the block interval, transaction throughput increases, the number of fork blocks increases, and network security becomes weak. Currently, the block interval of Ethereum is approximately 15 s, which is much longer than those of other blockchains. Therefore, both transaction throughput and the number of fork blocks are reduced; thus, the

stale block rate is also decreased, and network security is naturally stronger. Nevertheless, the block generation speed is very low.

Figure 11 illustrates the results of the tests conducted twice in Cases 1 and 2. Around the block interval of 10 s, in Case 1, the propagation time was 4.18 s, propagation rate was 96%, and stale block rate was 1.82%. In Case 2, the propagation time was 2.91 s, propagation rate was 91%, and stale block rate was 0.16%. As shown in Figure 11a, the propagation rate distribution ranged from 60% to 100% in the block interval range from 5 to 9 s for both cases. As shown in Figure 11b, the propagation rate fell to less than 5%, by approximately 6.8% that of Ethereum Mainnet, in the block interval range from 5 to 9 s. Therefore, in the Testnet, the block interval range from 5 to 9 s was used as the consensus time.



(**a**) Propagation Rate Distribution



(**b**) Stale Block Rate Distribution

Figure 11. Diffusion Distribution: (**a**) Propagation Rate Distribution; (**b**) Stale Block Rate Distribution.

In this study, such a problem was improved upon, and the block interval remained at approximately 5 s, which is a very fast speed for block generation. If the block interval increases, the number of forks increases owing to the trade-off, and network security weakens. However, if the stale block rate is maintained along with the increase in the number of forks and the block interval increases, the block propagation rate remains at 60–90%. Thus, there is no problem with a shorter block interval. In the current Ethereum consensus process, the stale block rate in the deciding subtree process is 5% or more in the block interval of 5 s or more. Therefore, there is no difference in the stable block rate from 1% to 6% in the current Mainnet.

5. Conclusions

The fundamental objectives of blockchain are to escape from the centralized server–client structure in which data are managed at one station and to prevent damage to distributed networks in which the same data are distributed to network participants. Therefore, a public blockchain network, which has the advantage of decentralization, employs PoW, a typical consensus algorithm, to verify the transactions between nodes. PoW has the original hash-based scalability issue. If this issue is addressed, PoW would be applicable to a blockchain application service. Given this background, although it is important to introduce a new consensus algorithm, it is necessary to improve the performance of the consensus algorithm without any detriment to decentralization, which is one of the applications of blockchain. Therefore, this study considered two cases. Even if the block interval is shorter than the current one, there are fewer security problems, even with a rise in stale block rate. In this manner, it is possible to solve the problem of an unnecessarily constant rise in the level of difficulty and the problem of an increase in H/W resources, which are disadvantages of PoW.

The experiment in this study was based on the current consensus algorithm of Ethereum, and there is no existing comparison between different consensus algorithms. The precondition of high decentralization with Ethereum is that the number of nodes is sufficiently large, as in the current Mainnet. Under these conditions, although the stale block rate is somewhat increased, no security problems occur. The block propagation rate was used as an index for scalability improvement in the block consensus process. With a change in the block propagation rate, the block interval was reduced. Therefore, the index was used to determine an improved block interval when the PoW consensus algorithm was redesigned. As security indexes of the block consensus process in the Ethereum PoW consensus algorithm, a stale block and block propagation rate were employed. By analyzing the changes in these parameters, this study showed that it is possible to shorten the block interval by influencing the three types of security issues. Although the experimental results were obtained in only two specific cases, continuous block-log data accumulation was made possible, as will be considered in future research work. With the accumulated results, it will be possible to determine the best block consensus time in a machine learning and decision-making process in future research. Data accumulation is the key resource for the improvement of PoW performance. Therefore, it is judged that the allowable rate (tolerance) of stable blocks will be used as an index to determine an improved block interval when the PoW consensus algorithm is redesigned.

Author Contributions: Conceptualization, H.K. and D.K.; Methodology, H.K.; Software, H.K.; Validation, H.K. and D.K.; formal analysis, H.K. and D.K.; investigation, H.K.; resources, H.K.; data curation, H.K.; writing—original draft preparation, H.K. and D.K.; writing—review and editing, H.K. and D.K.; visualization, H.K.; supervision, D.K.; project administration, H.K. and D.K.; funding acquisition, D.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This work was supported by the Korea Institute for Advancement of Technology (KIAT) grant funded by the Korea Government (MOTIE) (P0008691, HRD Program for Industrial Innovation).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System Bitcoin. 2009. Available online: https://bitcoin.org/en/bitcoin-paper (accessed on 26 August 2021).
2. Vujičić, D.; Jagodić, D.; Ranđić, S. Blockchain technology, bitcoin, and Ethereum: A brief overview. In Proceedings of the 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), Sarajevo, Bosnia and Herzegovina, 21–23 March 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6.
3. Zhou, Q.; Huang, H.; Zheng, Z.; Bian, J. Solutions to scalability of blockchain: A survey. *IEEE Access* **2020**, *8*, 16440–16455. [CrossRef]
4. Gilbert, S.; Lynch, N. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News* **2002**, *33*, 51–59. [CrossRef]
5. Chauhan, A.; Malviya, O.P.; Verma, M.; Mor, T.S. Blockchain and Scalability. In Proceedings of the 2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Lisbon, Portugal, 16–20 July 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 122–128.
6. Kim, S.; Kwon, Y.; Cho, S. A survey of scalability solutions on blockchain. In Proceedings of the 2018 International Conference on Information and Communication Technology Convergence, Jeju Island, Korea, 17–19 October 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1204–1207.
7. Irresberger, F.; John, K.; Saleh, F. *The Public Blockchain Ecosystem: An Empirical Analysis*; NYU Stern School of Business: New York, NY, USA, 2020.
8. Gervais, A.; Karame, G.O.; Wüst, K.; Glykantzis, V.; Ritzdorf, H.; Capkun, S. On the security and performance of proof of work blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 3–16.
9. Decker, C.; Wattenhofer, R. Information propagation in the bitcoin network. In Proceedings of the 2013 International Conference on Peer-to-Peer Computing, Trento, Italy, 9–11 September 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 1–10.
10. Karame, G. On the security and scalability of bitcoin's blockchain. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 1861–1862.
11. Ongaro, D.; Ousterhout, J. The Raft Consensus Algorithm. 2015. Available online: https://raft.github.io/slides/buildstuff2015.pdf (accessed on 26 August 2021).
12. Castro, M.; Liskov, B. Byzantine Fault Tolerance. U.S. Patent 6,671,821, 30 December 2003.
13. Castro, M.; Liskov, B. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.* **2002**, *20*, 398–461. [CrossRef]
14. Mingxiao, D.; Xiaofeng, M.; Zhe, Z.; Xiangwei, W.; Qijun, C. A review on consensus algorithm of blockchain. In Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017; pp. 2567–2572.
15. Tikhomirov, S.; Moreno-Sanchez, P.; Maffei, M. A quantitative analysis of security, anonymity and scalability for the lightning network. In Proceedings of the 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), 7–11 September 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 387–396.
16. Singh, A.; Parizi, R.M.; Han, M.; Dehghantanha, A.; Karimipour, H.; Choo, K.-K.R. Public Blockchains Scalability: An Examination of Sharding and Segregated Witness. In *Advances in Information Security*; Springer Science and Business Media LLC: Berlin, Germany, 2020; pp. 203–232.
17. Yu, G.; Wang, X.; Yu, K.; Ni, W.; Zhang, J.A.; Liu, R.P. Survey: Sharding in Blockchains. *IEEE Access* **2020**, *8*, 14155–14181. [CrossRef]
18. Gilad, Y.; Hemo, R.; Micali, S.; Vlachos, G.; Zeldovich, N. Algorand: Scaling byzantine agreements for cryptocurrencies. In Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, 28 October 2017; Association for Computing Machinery: New York, NY, USA, 2017; pp. 51–68.
19. Bentov, I.; Gabizon, A.; Mizrahi, A. Cryptocurrencies without Proof of Work. In Proceedings of the 2016 Conference of User Modeling, Adaption and Personalization, Halifax, NS, Canada, 13–17 July 2016; Springer International Publishing: New York, NY, USA, 2016; pp. 142–157.
20. Wang, Y. Another look at ALGORAND. *arXiv* **2019**, arXiv:1905.04463.
21. Li, X.; Jiang, P.; Chen, T.; Luo, X.; Wen, Q. A survey on the security of blockchain systems. *Future Gener. Comput. Syst.* **2020**, *107*, 841–853. [CrossRef]
22. Docker Compose Release Notes. Available online: https://docs.docker.com/compose/release-notes/ (accessed on 26 August 2021).
23. Github Repository. Available online: https://github.com/ethereum/go-ethereum/tree/1da33028ce88c4365d99471977098f4911fd38fa (accessed on 26 August 2021).
24. Go 1.9 Release Notes. Available online: https://golang.org/doc/go1.9 (accessed on 26 August 2021).