OPEN ACCESS

*International*
*Journal*
*Of Modern Engineering Research (IJMER)*

# Preprocessing Technique Based on Cloud for Lightweight Blockchain

Baikjun Choi[1], Jungwon Seo[2], and Sooyong Park[3]
[1]*CEO, Tilon Inc, Seoul, Republic of Korea.*
[2]*Department of Computer Science, Sogang University, Seoul, Republic of Korea*
[3]*Prof. Department of Computer Science, Sogang University, Seoul, Republic of Korea*
*Corresponding Author: Sooyong Park*

***ABSTRACT:****Two technologies with bright prospects in terms of potential development and market value are the Internet of Things (IoT) and blockchain. Although various studies have been conducted to converge the two technologies, the performance of IoT devices is not yet sufficient to use the current blockchain in IoT environments, due to the mathematical methods and mechanisms used in blockchain. Thus, making blockchain lightweight is essential for applying blockchain to IoT environments. This study proposes a blockchain platform structure where the desktop virtualization technology over Descktop as a Service (DaaS) is applied for lightweight blockchain, and a preprocessing technique is used by which an agent notifies blockchain nodes of the time that require consensus while blockchain nodes record data without consensus.*

***KEY WARDS:*** *Cloud, DaaS, Blockchain, Lightweight, BFT, PBFT*

---

Date of Submission: 11-04-2020 Date of acceptance: 27-04-2020

---

## I. INTRODUCTION

Blockchain is a data distribution storage technology in which users can record and query all data generated over a network. Market survey institutions predict that the development potential and market value of blockchain technology will rise steadily [1-2]. However, it is necessary to improve the performance of blockchain technology in order to apply blockchain to real life [3].

Various blockchain has been developed and utilized in many fields, due to their reputable technology, which supports a data repository that guarantees trust or trusted business [4-6]. In particular, many studies have been conducted on the fusion of blockchain with the Internet of Things(IoT) that is also considered as having a potential like blockchain [7-8]. A fusion of the IoT devices that are vulnerable to security issues by utilizing the security and transparency provided by blockchain [9] and can also provide an opportunity for blockchain to takes a leap forward.

However, blockchain uses complex mathematical methods and distribution mechanisms to protect their internal data and prevent outside attacks, requiring the use of many computer resources, while the performance of IoT devices is not sufficient to use existing blockchain technology in the IoT environment [10].

If IoT and blockchain converge, blockchain can be applied to various fields. For example, it is possible to make a monitoring system in hazardous workplace using IoT sensor devices with data integrity of blockchain. Also, it is possible to protect data for embedded devices of electric vehicles by blockchain. Thus, it is necessary to make blockchains lightweight to apply them to IoT environments.

This research proposes architecture where IoT devices are used by utilizing the desktop as a service (DaaS) technology for lightweight blockchain. It also proposes a preprocessing technique for lightweight blockchain by reducing the computer resources in Byzantine Fault Tolerance (BFT) consensus algorithm. In the preprocessing technique, an agent notifies a time that require consensus, while blockchain nodes record data without consensus. The agent accesses each node at a specific interval and notifies the node of the time when consensus is required, using two-step preprocessing. The first preprocessing step checks whether the recorded data are consistent in all nodes, and the second step notifies the node which data are not consistent and thereby require consensus.

The experiment in this paper presents results of DaaS performance by calculating capacity for transmission per frame. Also, it presents results where the preprocessing technique is applied to the Practical Byzantine Fault Tolerance (PBFT) consensus algorithm, which is the most popular among BFT algorithms in the private blockchain environment.

This paper is organized as follows. Section 2 presents related work for this research. In section 3, the desktop virtualization-based blockchain and the process of the preprocessing-based consensus algorithm are described. Section4, and 5 present experiments, conclusions, respectively.

## II. RELATED WORK

This section introduces studies on a consensus algorithm for lightweight blockchain. The studies related to the present study include Microchain [11], Lightweight [12], Proof of Sincerity [13], Storage Compression Consensus (SCC) [14], and Proof of Authentication [15].

Microchain converge proof of credit and voting based chain finality techniques to propose a lightweight consensus algorithm that could be run in IoT environments. In Lightweight [12], a modified consensus algorithm of proof of stake (PoS) authorizes block generation based on how steadily an existing block generator attempted to create blocks to achieve lightweight blockchains. In Proof of Sincerity, a new consensus algorithm was proposed to solve the difficulty of running existing proof of work (PoW) in IoT devices, such as mobile phones, due to excessive computer resource consumption. In SCC, a consensus algorithm was proposed by compressing blocks to solve the storage capacity limit of IoT devices. In Proof of Authentication, a new form of consensus algorithm whose hash function size was reduced as a modification of PoW was proposed to run the consensus algorithm even in IoT environments.

These studies [11-15] researched consensus algorithms for lightweight blockchain using various methods. However, they were limited in that they only modified existing consensus algorithms such as PoS or PoW. They were also limited in that they lacked analysis and the proof of safety and robustness of the consensus algorithm, which may occur due to the modification of existing consensus algorithms.

In contrast, this paper does not modify existing consensus algorithms but proposes a lightweight blockchain method by minimizing the resources required for consensus through preprocessing before consensus.

## III. APPROACH

### *3.1 Desktop virtualization-based blockchain platform*

The DaaS technology is a computer virtualization technology for cloud services. When a cloud is constructed using DaaS technology, users have an advantage in that they can use their personal computers anywhere, anytime, regardless of the machine performance. When a blockchain is constructed utilizing the DaaS technology, IoT devices are connected to the cloud, thereby performing the roles of blockchain nodes. Figure1 shows the flow of the cloud-based blockchain network using a DaaS server.

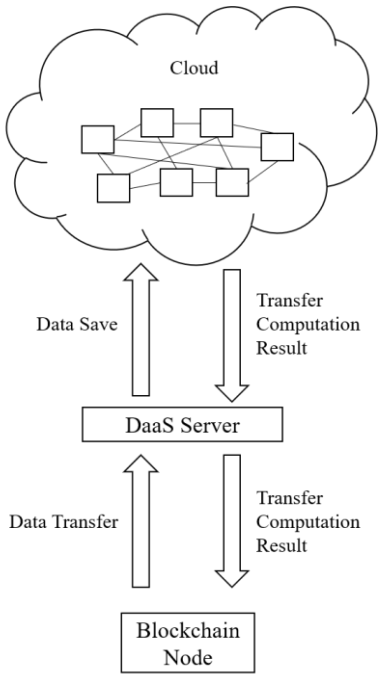


**Figure 1: Cloud-based blockchain network through the DaaS server**

The blockchain nodes contained in the blockchain network transfer data that are preferred to be saved to the DaaS server. The DaaS server then forwards the transferred data to the cloud, where blockchain network-related computation is processed. After this only the computation result is delivered to blockchain nodes.

Figure 2 shows the structure of the DaaS server and the blockchain node used in the cloud, as proposed in this studying. By proposed Daas Server architecture, IoT devices can connect to cloud server more efficiently.

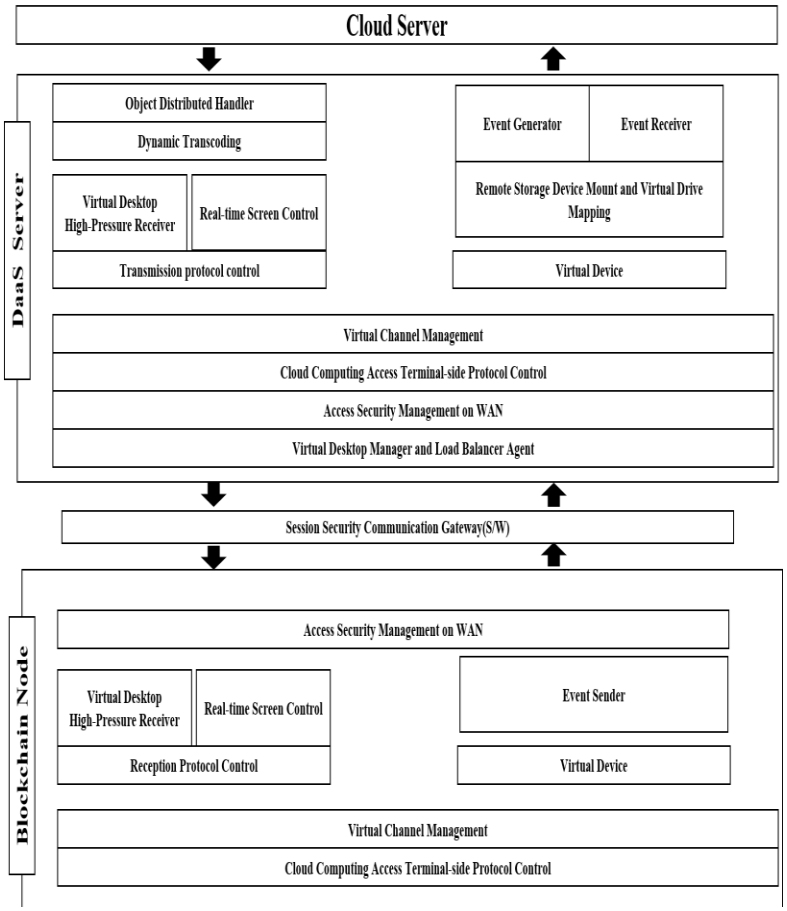


**Figure 2: Structure of DaaS server and Blockchain node**

Figure 2 shows the structure of the DaaS server and the blockchain node used in the cloud, as proposed in this study. The display area real-time control unit in the DaaS server side separates files into text, image, and video and employs a compression technique according to the attribtue of the file type. The object processor is responsible for collecting information of transferred files. The functions of the blockchain nodes consists of basic functions for communication with the DaaS server. The virutal Channel management unit included in the structure of the DaaS server and blockchain node plays a role in matching the information of the transferred files.

### *3.2Preprocessing technique*

This section describes the preprocessing technique utilizing a cloud-based blockchain network using the DaaS server proposed in the previous section

#### *3.2.1Agent*

An agent in this research is defined as an object to help achieve consensus. The agent notifies of the moment when consensus is needed and participates in the overall process that generates blocks if needed after performing consensus. The agent is an independent object that is separate from the blockchain network and is responsible for the preprocessing process. The agent does not run at normal times, but it starts by an arbitrary node elected at the defined block generation interval. Although the agent is an independent object, it cannot decide alone what influences blockchain; all processing results can be broadcasted to blockchain network after obtaining the signature from the key node. The agent temporarily stores all data and deletes the temporarily stored data immediately as soon as the preprocessing and consensus process is complete, thereby preventing data loss and any hacking threat against the agent.

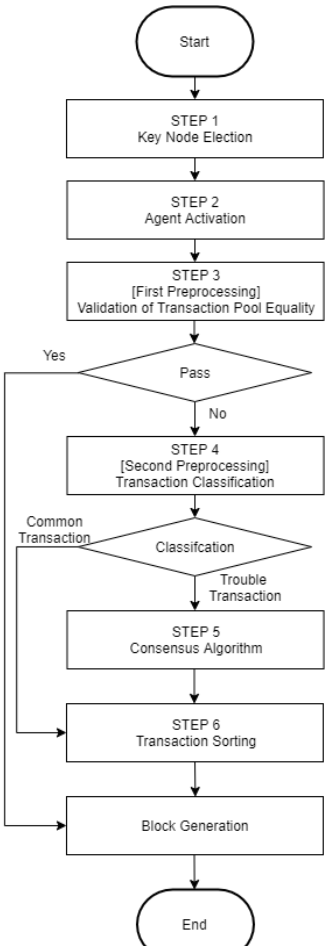


**Figure 3: Overall flow of the agent-based preprocessing technique**

***3.2.2Key node election***

This study assumes that a unique identification number is assigned to a blockchain node when the node participates in the network for the first time. The algorithm for the key node is expressed by the pseudo code in Figure 4. Each of the nodes uses a hash value of the previous block at a certain interval (4) as a seed to elect an arbitrary node (6) through the random seed algorithm. The hash value of the previous block means that all nodes can acquire the same arbitrary number, because all nodes agree already. The key node at a specific round is identified through the acquired arbitrary number (7). The node elected as the key node makes a signature by encrypting its own unique identification number along with the hash value of the previous block with the private key (8), and it transfers the signature, which is tied with the public key made by its own private key to the agent (9).

**Algorithm 1** Key Node Election

1: $Max_{range}$ = number of blockchain node
2: $t$ = block generation interval
3: $Seed$ = previous block hash
4: **if** Timer = $t$ **then**
5: $Set_{range} = 1$ *to* $Max_{range}$
6: $Result_{random}$ = random algorithm $(Set_{range}, Seed)$
7: **if** $Node_{idNumber} = Result_{Random}$ **then**
8: $Key_{sig} = Key_{pri}\ (Node_{idNumber}, Seed)$
9: send to Agent $(Key_{sig}, Key_{pub})$
10: **end if**
11: **end if**

**Figure 4: Pseudo code of Key node election**

***3.2.3Activation of agent through key node***

Figure 5 shows the pseudo code of agent activation by the key node. The agent does not activate until it receives a signature from the key node. The agent receives a message from the key node and performs the verification of whether the signature transferred by the key node and the public key of the key node are correct (1). The agent does not activate either until the correct signature and public key are transferred, the result value of the random seed algorithm is different from the unique identification number of the key node, or the transferred signature and public key are not sent by the key node elected by the random seed algorithm (6). Once the key node verification is successfully complete, the notification of agent activation is sent to all nodes (3) and enters the preprocessing process (4).

**Algorithm 2** Agent Activation

1: **if** $Key_{sig}, Key_{pub} \in Key_{node}$ **then**
2: $Agent_{state}$ = ON
3: broadcast $(Agent_{state})$
4: $first\ Preprocessing\ (TxPool_n)$
5: **else**
6: $Agent_{state}$ = OFF
7: **end if**

**Figure 5: Pseudo code of agent activation**

***3.2.4First preprocessing***

The first preprocessing process aims to omit the existing consensus process in progress to receive confirmation from participants even if consensus is not needed.

The blockchain nodes that receive a message about the agent is activated disclose the transaction pool, a repository that stores self-verified transactions and all previously recorded data, to the agent. As shown in Figure 6, the agent accesses each of the nodes to fetch the transaction pool, sorted by a certain criterion (1), and it converts the transaction pool into a hash form using the hash function (2). Because of the hash function characteristic, if nodes that participate in the network store the same data in the transaction pool, they will have the same hash value, and if any node stores different data in the transaction pool, that node will have a different hash value. If all nodes have the same transaction pool hash value (4), the agent verifies the preprocessing result at the key node (5). Once the key node verification is successfully completed, the agent transfers the result value along with the key node signature to other nodes (7). Thus, nodes generate blocks without the need to use the consensus algorithm, based on the transferred result value. If any node has a different transaction pool hash value, the agent performs the second preprocessing (10).

**Algorithm 3** First Preprocessing

1: access $TxPool_n$
$= \{transaction_1,\ transaction_2,\ \cdots\ transaction_i\}$
$\leftarrow TxPool_n \in Node_n$
2: hash algorithm $(TxPool_n)$
3: add $Hash_{txPooln}$ *to* $Hash_{result}$
$= \{Hash_{txPool1},\ Hash_{txPool2},\ \cdots\ Hash_{txPooln}\}$
4: **if** $Hash_{result}$ elements are same **then**
5: send to $Key_{node}$ $(Result_{p1})$
6: **if** $Result_{p1}$ = valid **then**
7: broadcast $(Result_{p1})$
8: **end if**
9: **else**
10: $Second\ Preprocessing\ (TxPool_n)$
11: **end if**

**Figure 6: First preprocessing pseudo code**

***3.2.5Second preprocessing***

The second preprocessing is a transaction classification task to find a transaction subject to consensus if the nodes do not have the same data in the first preprocessing.

Figure 7 shows the algorithm. The agent classifies transactions utilizing the transaction pool fetched from existing nodes. If transactions are identified by all nodes, they are classified as common transactions (2) and are

temporarily saved in the agent until the block is generated (3). Transactions other than common transactions are classified as trouble transactions (5).

**Algorithm 4** Second Preprocessing

1: **if** $transaction_i = Common_{transaction}$ **then**
2: add to $C_t$
$= \{Common_{transaction1}, \cdots Common\ transaction_i\}$
3: save $C_t$ until block generate
4: **else**
5: add to $T_t$
$= \{Trouble_{transaction1}, \cdots Trouble_{transactioni}\}$
6: execute consensus algorithm with $T_t$
7: **end if**

**Figure 7: Second preprocessing pseudo code**

### *3.2.6Consensus of trouble transaction*

In this step, the consensus process is performed through the trouble transactions classified in the previous step. Any consensus algorithm can be applied in this process. Transactions whose consensus is complete are classified as consensus transactions. Trouble transactions whose consensus is not achieved in this process are completely removed.

### *3.2.7Transaction sorting*

The transaction sorting algorithm is expressed by the pseudo code in Figure 8. The agent sorts the common transactions classified in the second preprocessing and consensus transactions gained through the consensus algorithm (1). This process may utilize the unique sorting techniques provided by blockchain. This study sorts trans-actions by timestamps (2) and transfers the results to the key node (3). The key node validates the value transferred from the agent by signing a signature. The agent that receives the signed value sends it to the blockchain nodes (4).

**Algorithm 5** Sorting

1: $Temp\_Result_{p2} = \{C_t \cup Consensus_t\}$
2: $Result_{p2} =$ sort $(Temp\_Result_{p2})$
3: send to $Key_{node}(Result_{p2})$
4: **if** $Result_{p2} =$ valid **then**
5: broadcast $(Result_{p2})$
6: **end if**

**Figure 8: Transaction sorting pseudocode**

## IV. EXPERIMENTS

This section presents the experiment results of DaaS performance measurement to show how the use of this preprocessing technique processes consensus more quickly than existing techniques.

### *4.1. Performance analaysis of DaaS*

Table 1 calculates the volumes of text, image, and video required for transmission per frame, using the DaaS server architecture in this study.

***Table I.*Performance evaluation of DaaS**

| Category | Text | Webpage | Image | Graphic | Video | 3D video |
|---|---|---|---|---|---|---|
| **HuffYuv (YV12,12bit)** | 439948 | 472504 | 919748 | 841692 | 907784 | 551280 |
| **HuffYuv (YV12,12bit)** | 936052 | 1047216 | 1852136 | 1870600 | 1884226 | 1224461 |
| **Zlib (YV12, 12bit)** | 131303 | 282158 | 979410 | 841604 | 997655 | 834647 |
| **Zlib (RGB,32bit)** | 191401 | 525547 | 2567978 | 2230376 | 2754009 | 2448079 |
| **JPEG (RGBA,32bit)** | 213905 | 245809 | 214881 | 197054 | 213378 | 174484 |
| **h.264** | 209~12768 | 2231~11175 | 235~9270 | 242~8288 | 354~34131 | 208~24155 |

A mobile phone was used as an IoT device to process the experiment. The experimental results verified that there was no significant performance degradation, because selective compression methods were applied when connecting to the cloud using the DaaS server. In particular, the highest performance was measured when sending contents using JPEG and h.264 codec.

### *4.2. Performance analaysis of preprocessing technique*

This section conducted an experiment to verify how much the consensus speed changed due to the reduction in computer resources required for blockchain consensus when applying the preprocessing technique to the PBFT consensus algorithm. A transaction was generated every 10ms for the experiment. The block was assumed to be generated at 10 sec intervals. Malicious transactions were generated randomly, in accordance with the set ratio of malicious transactions, to reduce entry to the trouble transaction consensus process. The number of nodes was set to 4 to 100, which increased with increments of 4 nodes. The PBFT algorithm was measured in an environment without malicious transactions. To compare the processing speed between the preprocessing technique proposed in this study and PBFT, the processing speed of the algorithm whose malicious transaction rate was 33% was also measured when the fault rate was 1/3, which was the maximum allowable fault rate in the PBFT algorithm.

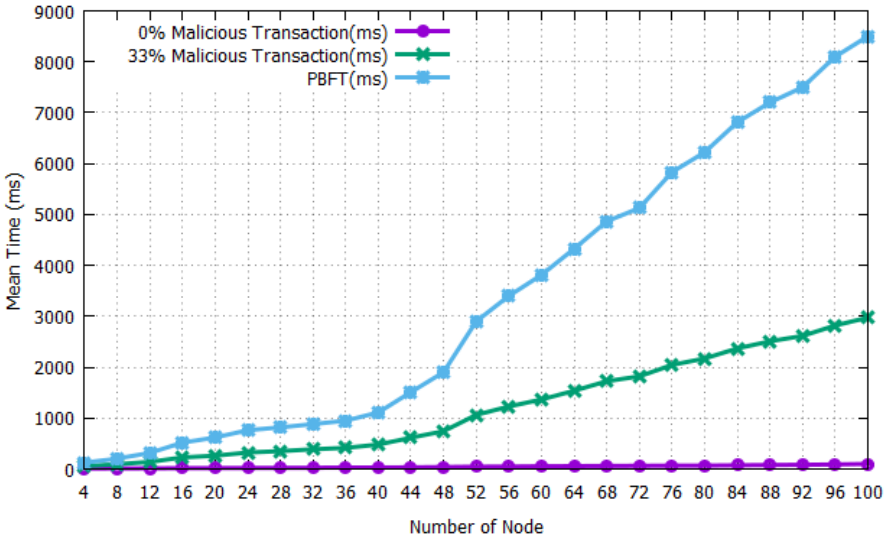


**Figure 9: Processing time according to the number of nodes and malicious trnsactions**

Overall, the consensus time gradually increased, because the computation to be processed with the increasing number of nodes also increased. The graph in Figure 9 shows that when the rate of malicious transactions is 0%, there seems to be no significant speed change. However, since the transaction pool hash values to be tested increase with the increasing number of nodes, the processing speed gradually increases. When there are no malicious transactions, the processing speed is the fastest, because only transaction pool tests which is the first preprocessing step are processed, as shown in Figure 9. The experimental results also verify that even if the number of malicious transactions is 33% of total transactions, the proposed algorithm shows faster processing than the PBFT algorithm by about 39%, without malicious node. The existing PBFT must perform consensus continuously to validate correct transactions from other nodes, thereby steadily increasing computation, resulting in a slowing down of IoT devices due to excessive resource usage. But the preprocessing technique minimizes the resource usage of devices in the consensus process, resulting in the measurement of faster speed.

### *4.3. Scalability analysis of preprocessing technique*

This section shows results of extra experiment about scalability limits of preprocessing technique when using PBFT algorithm.

$$Expected\ Time = p_1 + f(p_2 + cs) \quad \text{.....(1)}$$

$p_1$ indicates the processing time of the first preprocessing step which is Figure 6. $p_2$ indicates the processing time of the second preprocessing step which is Figure 7. $cs$ is consensus algorithm processing time and $f$ is the percentage of malicious transactions.

Equation 1 describes the average expected time when preprocessing technique applied. Average expected time refers to the average processing time taken when the technique runs an infinite process until one block is created.

***Table II.*** Processing Time of Preprocessing Step and PBFT

| Number of Nodes | $p_1(ms)$ | $p_2(ms)$ | $cs(ms)$ |
|---|---|---|---|
| **4** | 8.260522 | 39.82271 | 123.7411 |
| **8** | 11.220054 | 68.1713 | 204.891 |
| **12** | 13.427149 | 100.6671 | 316.0375 |
| **16** | 15.539575 | 189.315 | 517.615 |
| **20** | 18.4699 | 191.5774 | 618.661 |
| **24** | 19.474479 | 253.32612 | 762.3018 |
| **28** | 21.0823 | 283.11001 | 817.689 |
| **32** | 24.253593 | 327.8439 | 880.9612 |
| **36** | 28.192941 | 336.1532 | 950.3651 |
| **40** | 29.004264 | 393.55129 | 1110.36941 |
| **44** | 32.04488119 | 432.2858 | 1510.351 |
| **48** | 38.70372475 | 443.44086 | 1900.636 |
| **52** | 45.235952 | 491.6951 | 2900.321 |
| **56** | 48.38571429 | 532.74596 | 3400.324 |
| **60** | 53.553152 | 561.50059 | 3812.631 |
| **64** | 57.3364123 | 605.3641 | 4325.91 |
| **68** | 60.20151 | 681.3914 | 4865.021 |
| **72** | 62.429245 | 731.12813 | 5125.561 |
| **76** | 65.458039 | 766.3312 | 5831.12 |
| **80** | 67.1571 | 791.7781 | 6221.23 |
| **84** | 72.982313 | 842.1971 | 6812.23 |
| **88** | 78.428301 | 901.3951 | 7200.31 |
| **92** | 81.978102 | 941.3789 | 7500.21 |
| **96** | 88.792111 | 1002.3197 | 8100.32 |
| **100** | 99.6611 | 1081.3312 | 8500.24 |

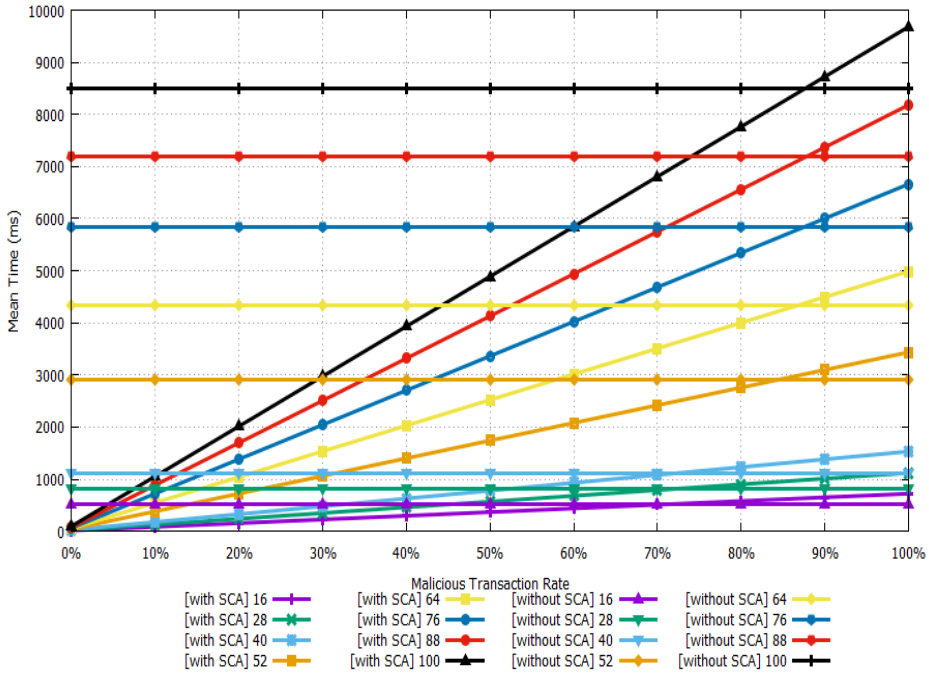


**Figure 10: Expected time by malicious node rate (Formula)**

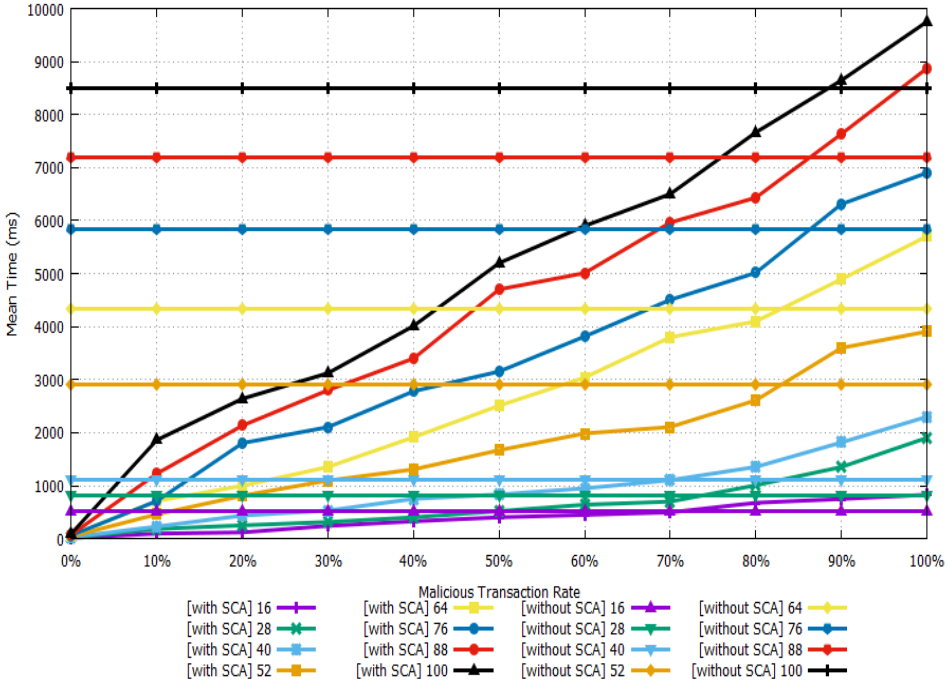


**Figure 11: Processing time by malicious node rate (Experiment)**

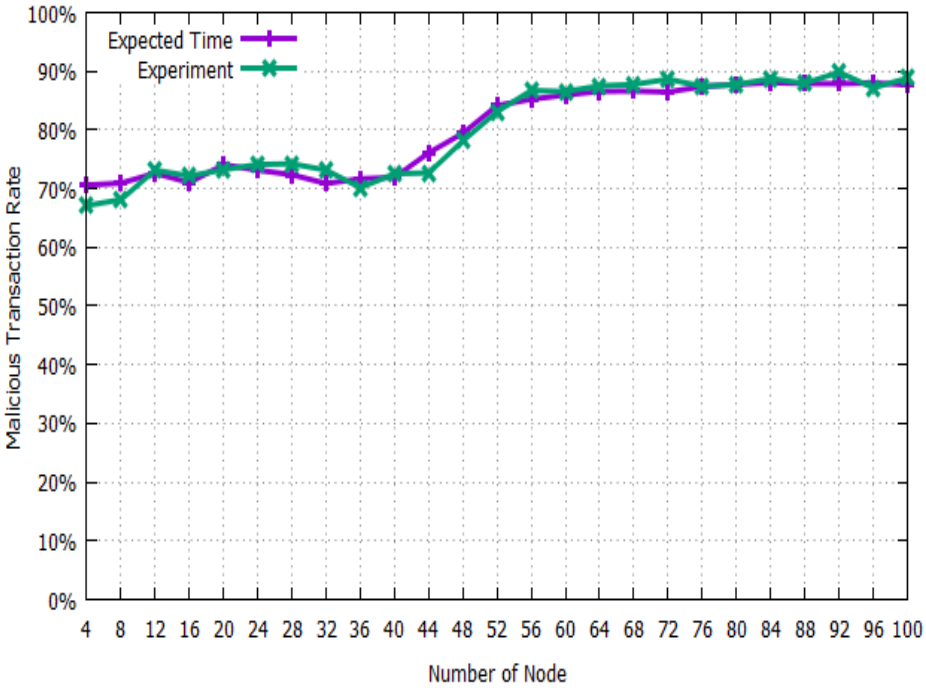


**Figure 12: Scalability by malicious node rate**

Figure 10 shows the expected time that is calculated from the data in Table 2. Figure 11 shows the average processing time from the actual experiment. Both graphs represent processing time as the proportion of malicious transactions increases. Graphs shown in straight form (without SCA) refer to the processing time of the PBFT algorithm without the preprocessing technique is applied.

Because PBFT performs consensus each time regardless of the percentage of malicious transactions, the processing time appears to be the same. The graphs shown in increasing form (with SCA) show the average processing time when the preprocessing algorithm is applied. The results of Figure 10 and Figure 11 visually indicate that the expected results from the Equation1 are not much different from the results the results of the direct experiment. Figure 10 and 11 express only eight nodes (16, 28, 40, 52, 64, 76, 88, 100) for visibility of

graphs. When compared using all nodes, the standard deviation of the two graphs which are Figure 10 and 11 was 108.8586ms. Figure 12 indicates connection of interception points where the straight line and increasing line meet from figure 10 and 11. In Figure 12, upper range of the graph means that the PBFT algorithm processing time is faster than preprocessing technique applied. For example, if the ratio of malicious transactions is about 84 or more in an environment with 52 nodes, using only PBFT algorithm is faster than using preprocessing technique. Conversely, the lower range of the graph represents an environment in which the processing speed using the preprocessing technique is faster than using only PBFT algorithm. Thus, when the preprocessing technique is applied to PBFT consensus algorithm, it allows for a malicious transaction ratio of approximately 68 to 70 percent in environments with number of nodes in the network is small, and about 80 to 85 percent in environments with number of nodes are big.

## V. CONCLUSION

This paper proposed a method that using DaaS technology and a preprocessing technique for lightweight blockchain.

It proposed a measure to make a blockchain network by easily connecting IoT devices whose volume was relatively small to the cloud using DaaS technology, and it conducted an evaluation to determine the performance when accessing the DaaS from mobile phones.

This paper also proposed a preprocessing technique utilizing an agent, thereby presenting a measure to reduce the resources required for consensus in blockchain environments. The agent determines whether or not consensus is required by transactions collected by vising blockchain nodes, and it generates blocks after performing consensus, if needed. By doing this, consensus is not frequently needed, thereby reducing the resources required for consensus and resulting in a fast consensus speed. When an experiment was conducted by applying the preprocessing technique to the PBFT, the preprocessing-applied consensus algorithm was verified to be 39% faster on average than that of the PBFT algorithm without malicious nodes, even with a 33% rate of malicious transaction generation.

However, this study had a limitation in that if the DaaS was employed, the computer resources consumed in IoT devices varied whenever the performance of the DaaS server changed. It also lacked a means of defense against attacks that occurred inside or outside blockchains, targeting the agent when using the preprocessing technique. For future study, additional experiments will be conducted to overcome the above drawbacks, and blockchain platforms will be studied where various IoT devices can be connected, for example sensor devices rather than IoT devices in a cloud environment.

## REFERENCES


[1]. Deep Shift Technology Tipping Points and Societal impact. Available online: http://www3.weforum.org/docs/WEF_GAC15_Technological_Tipping_Points_report_2015.pdf (accessed on 10 March 2020)

[2]. Blockchain Market Shares, Market Strategies, and Market Forecasts, 2018 to 2024. Available online: https://www.ibm.com/downloads/cas/PPRR983X (accessed on 3 March 2020)

[3]. Zibin Zheng; Shaoan Xie; Hong-Ning Dai; Xiangping Chen. Blockchain challenges and opportunities: a survey. *Int.J. Web and Grid Services* 2018, *Vol.14*, pp.352-375

[4]. Kari Krpela; Jukka Hallikas; Tomi Dahlberg. Digital Supply Chain Transformation toward Blockchain Integration. Proceedings of the 50th Hawaii Internation Conference on System Sciences, Hawail, U.S.A, 04 January 2017

[5]. Lin William Cong; Zhiguo He. Blockchain Disruption and Smart Contracts. *The Review of Financial Studies*, 2019, *Vol.32*, pp.1754-1797

[6]. Steve Huckle; Rituparna Bhattacharya; Martin White; Natalia Beloff. Internet of Things, Blockchain and Shared Economy Applications, *Procedia Computer Science* 2016, *Vol.98*, pp.461-466

[7]. Kolbeinn Karlsson; Weitao Jiang; Stephen Wicker; Edwin Ma; Robbert van Renesse; Hakim Weatherspoon. Vegvisir: A Partition-Tolerant Blockchain for the Internet-of-Things. 2018 IEEE 38th International Conference on Distributed Computing System, Vienna, Austria, 02 July 2018

[8]. Joshua Ellul; Gordon J. Pace. AlkylVM: A Virtual Machine for Smart Contract Blockchain Connected Internet of Things. 2018 9th IFIP International Conference on New Technologies, Mobility and Security, Paris, France, 26 February 2018

[9]. Nir Kshetri. Can Blockchain Strengthen the Internet of Things? *IT Professional* 2017, *Vol.19*, pp68-72

[10]. Yinqiu Liu; Kun Wang; Yun Lin; Wenyao Xu. LightChain: A Lightweight Blockchain System for Industrial Internet of Things. *IEEE Transactions on Industrial Informatics* 2019, *Vol.15*, pp.3571-3581

[11]. Ronghua Xu; Yu Chen; Erik Blasch; Genshe Chen. Microchain: A Hybrid Consensus Mechanism for Lightweight Distributed Ledger for IoT. *ArXiv* abs/1909.10948 (2019)

[12]. Jan Hackfeld. A lightweight BFT consensus protocol for blockchains. *ArXiv* abs/1903.11434

[13]. Miraz Uz Zaman; Tong She; Manki Min. Proof of Sincerity: A New Lightweight Consensus Approach for Mobile Blockchain. 2019 16th IEEE Annual Consumer Communications & Networking Conference, Las Vegas, USA, 11 January 2019

[14]. Teasung Kim; Jaewon Noh; Sunghyun Cho. SCC: Storage Compression Consensus for Blockchain in Lightweight IoT Network, 2019 IEEE International Conference on Consumer Electronics, Las Vegas, USA, 11 January 2019

[15]. Deepak Puthal; Saraju P. Mohanty. Proof of Authentication: IoT-Friendly Blockchains, *IEEE Potentials* 2019, *Vol.38*, pp.26-29

**Acknowledgement**

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program( IITP-2020-2017-0-01628 ) supervised by the IITP(Institute for Information & communications Technology Promotion)

Sooyong Park."Preprocessing Technique Based on Cloud for Lightweight Blockchain." *International Journal of Modern Engineering Research (IJMER),* vol. 10(03), 2020, pp 16-26.