electronics

Article

Fusion Chain: A Decentralized Lightweight Blockchain for IoT Security and Privacy

Dongjun Na and Sejin Park *

Department of Computer Engineering, Keimyung University, Daegu 1095, Korea; 5544616@stu.kmu.ac.kr
* Correspondence: baksejin@kmu.ac.kr; Tel.: +82-53-580-5270

Abstract: As the use of internet of things (IoT) devices increases, the importance of security has increased, because personal and private data such as biometrics, images, photos, and voices can be collected. However, there is a possibility of data leakage or manipulation by monopolizing the authority of the data, since such data are stored in a central server by the centralized structure of IoT devices. Furthermore, such a structure has a potential security problem, caused by an attack on the server due to single point vulnerability. Blockchain's, through their decentralized structure, effectively solve the single point vulnerability, and their consensus algorithm allows network participants to verify data without any monopolizing. Therefore, blockchain technology becomes an effective solution for solving the security problem of the IoT's centralized method. However, current blockchain technology is not suitable for IoT devices. Blockchain technology requires large storage space for the endless append-only block storing, and high CPU processing power for performing consensus algorithms, while its opened block access policy exposes private data to the public. In this paper, we propose a decentralized lightweight blockchain, named Fusion Chain, to support IoT devices. First, it solves the storage size issue of the blockchain by using the interplanetary file system (IPFS). Second, it does not require high computational power by using the practical Byzantine fault tolerance (PBFT) consensus algorithm. Third, data privacy is ensured by allowing only authorized users to access data through public key encryption using PKI. Fusion Chain was implemented from scratch written using Node.js and golang. The results show that the proposed Fusion Chain is suitable for IoT devices. According to our experiments, the size of the blockchain dramatically decreased, and only 6% of CPU on an ARM core, and 49 MB of memory, is used on average for the consensus process. It also effectively protects privacy data by using a public key infrastructure (PKI).

Keywords: blockchain; internet of things; inter planetary file system; public key infrastructure; practical byzantine fault tolerance

Citation: Na, D.; Park, S. Fusion Chain: A Decentralized Lightweight Blockchain for IoT Security and Privacy. *Electronics* **2021**, *10*, 391. https://doi.org/10.3390/electronics10040391

Academic Editors: Javier Prieto and Fernando De la Prieta
Received: 30 December 2020
Accepted: 29 January 2021
Published: 5 February 2021

1. Introduction

The IoT (internet of things) is a technology that connects everything to the Internet, from small communication objects, like embedded sensors, to large machines, like vehicles. By connecting Internet functionality, IoT devices have lots of advantages, such as higher manageability and reachability. The number of IoT devices is expected to increase from 30.73 billion in 2020 to 75.44 billion in 2025 [1]. With the increase in utilization, the scope of IoT has become widely spread to various fields, such as smart factories, transportation, energy, wholesale, city, healthcare, supply chain management, livestock, and construction [2]. In addition, IoT devices will generate a total data of 79.4 ZB in 2025 [3]. In the meantime, the security of IoT has become more important as it produces big data and collects sensitive information, such as personal biometrics, images, photos, and voices [4]. However, the existing IoT devices use a centralized architecture to store and process data. They collect and relay lots of data to a central server, which causes the problem of monopolizing data rights. Furthermore, there are also lots of adversary models, such as the Mirai botnet

method [5], that attack the central server by infecting IoT devices, using the vulnerability of the use of the central network method of IoT. Such adversary models have actually hacked 500,000 servers, home routers, and IoT devices, and there have been cases in which user passwords stored in a central server have been leaked [6]. Therefore, the centralized architecture of IoT devices becomes a security problem.

The cryptography technology used for the security of the existing centralized architecture cannot guarantee the reliability of the data because the trusted third party performs data forgery prevention. In addition, the distributed file system has the advantage of being able to cope with single point errors in which multiple nodes distribute and store a file, but cannot cope with data forgery. In the case of distributed systems, such as Ceph and GlusterFS, a copy maintenance method such as N-way replication, etc., and a method of restoration when a fault occurs, using an erasure code, is used. However, this method does not guarantee the authenticity of the data. Moreover, since restoration is possible only in specific error conditions, this system alone cannot guarantee the reliability and integrity of the data. Blockchain technology has been discussed as an effective solution to this problem. Blockchain has all the advantages of such a system, and being based on a consensus algorithm, network nodes can guarantee the reliability of data and prevent forgery, thus effectively solving the preceding problems. In brief, the main idea is that the blockchain network uses a decentralized method based on a P2P network, where nodes can participate without adding or changing additional equipment, and the blockchain is distributed and stored for each node with the same blockchain data. Therefore, even if a problem occurs in some devices, the overall system is less affected. Due to the characteristics of these blockchains, a blockchain-based IoT network has some advantages. First, data is stored for each node of the blockchain network, making it difficult to manipulate data for adversary purposes. Second, it effectively defends against attacks against servers, such as DDoS, due to the decentralized structure [7]. Third, even if a problem occurs in some devices due to the connection between devices, the overall system is less affected. With these advantages a blockchain based IoT is a good solution for various security problems in the IoT environment.

1.1. Blockchain Requirement and IoT Device Specification

Table 1 shows a list of minimum requirement for representative blockchain platforms, and Table 2 shows a specification of an existing IoT hardware platform. Briefly, a blockchain node requires a PC-level hardware specification, but IoT platforms do not support such a high specification. As described in Table 1, Bitcoin and Ethereum Full Node require at least 200 GB and 465 GB of disk space, respectively. This is clearly not suitable for IoT devices, which have a very limited size of flash storage, as described in Table 2. In addition, Bitcoin and Hyperledger Fabric require a minimum 1 GB and 4 GB of memory, respectively. Note that most IoT devices have a KB–MB size memory in Table 2.

Table 1. Minimum requirements for representative blockchain platforms.

Blockchain	Disk Space (Least)	Memory (Least)	System	Operating System
Bitcoin Core [8]	200 GB	1 GB	Laptop ARM chipsets 1GHz	Mac OS, Linux, Windows 7/8.x/10
Ethereum Full-Node	464 GB	-	Laptop ARM chipsets	Linux, MacOS
Hyperledger Fabric	-	4 GB	-	Mas OS, Ubuntu Linux

Table 2. Comparison of the existing internet of things (IoT) supported hardware platforms [9].

Platform	CPU	GPU	Clock Speed	Memory	Storage (Flash)
Intel Galileo Gen2	Intel®Quark™ SoC X1000	-	400 MHz	256 MB	8 MB
Intel Edison	Intel®Quark™ SoC X1000	-	100 MHz	1 GB	4 GB
Beagle Bone Black	Sitara AM3358BZCZ100	PowerVR SGX530 @520 MHz	1 GHz	512 MB	4 GB
Electric Imp 003	ARM Cortex M4F	-	320 MHz	120 KB	4 Mb
Raspberry Pi B+	Broadcom BCM2835 SoC based ARM11 76JZF	VideoCore IV®Multimedia@ 250 MHz	700 MHz	512 MB	SD Card
ARM NXP LPC1768	ARM Cortex M3	-	96 MHz	32 KB	512 KB

1.2. Challenges

Blockchain technology is suitable as a solution for the security of IoT, but due to the size of the blockchain, computational power, and transparency of data, it has the following problems when applied to IoT devices.

(1) Blockchains increase in size as time goes by, and require a large amount of storage; to maintain a full node, the size of the Ethereum blockchain has reached 308 GB, and 271 GB in the case of Bitcoin, to date [10].
(2) The average performance of IoT devices is low. The IoT device's CPU performance is not suitable for participation in the consensus algorithm of the existing blockchain technology. The PoW consensus process recommends GPU devices, and requires a lot of computation power.
(3) As the use of IoT devices increases, IoT devices store sensitive information, such as personal bioinformatics, photos, and images, and it becomes a serious privacy problem once they are leaked by attacks. In the case of TRENDnet, the company that produces and sells SecurView, due to a security problem with its IoT products, lots of images inside the homes of about 700 households were leaked to hackers [11].

Blockchains optimized for IoT devices that solve these problems, could solve the security problems of existing IoT devices.

1.3. Research Contribution

In this paper, we propose a lightweight blockchain named Fusion Chain that supports IoT devices using low CPU processing power, and improves security by orchestrating IPFS (interplanetary file system) [12], PBFT (practical Byzantine fault tolerance) [13], and PKI (public key infrastructure) [14]. Therefore, Fusion Chain effectively supports IoT devices by solving problems (1), (2), and (3) in Section 1.2. The methods for optimization for IoT devices are as follows.

- In order to solve the capacity problem, Fusion Chain leverages IPFS (interplanetary file system) as a backend universal storage. IPFS is a distributed file system using a P2P network. In Fusion Chain, the size of the blockchain is reduced by migrating the blocks to IPFS. Instead, it maintains a 32-byte IPFS hash block.
- The PBFT (practical Byzantine fault tolerance) consensus algorithm is converged to overcome the low computing power of the IoT. PBFT is suitable for IoT devices by using network-based consensus rather than computing power by sending a request through the network's broadcasting process, and then counting the responses.
- PKI (public key infrastructure) solves the problem of revealing private data to all blockchain nodes. PKI consists of a private key stored by itself and a public key disclosed to the other party, so data privacy can be guaranteed by encrypting data with the public key of the user who owns the IoT device, and then decrypting it with the user's private key only.

The remaining part of this paper will introduce related work and explain background knowledge on IoT and blockchain specifications. After that, the architecture and implementation details are described. Last, experimental will be shown and concludes this paper.

2. Related Work

2.1. Lightweight Blockchain

In Sensor-chain [15], a lightweight solution was proposed for the appropriate use of IoT devices in a blockchain. A spatial blockchain that divides the blockchain into space units and a migration manager function were used, in conjunction with migration over time. Each node owns a blockchain according to its space, and the content of the blockchain data is aggregated into one block, the size of the blockchain is reduced by starting from that block. Although this approach reduces blockchain size for IoT devices, it also lost blockchain data. In contrast Fusion Chain maintains all data by migrating to IPFS.

IOTA [16] is a cryptocurrency platform designed to apply blockchain to IoT. The size of the blockchain was reduced and made suitable by using a proprietary blockchain tangle algorithm. However, due to the semi-centralized form of the IoT, which does not achieve complete decentralization of the blockchain, the node storing a lot of data can be attacked, and there is a vulnerability [17] to hacking methods such as a centralized attack. In addition, it is not suitable for IoT devices with low computation power, since it uses PoW as the consensus algorithm, which requires high computation power.

In IPFS for Smart Contract [18], a solution using IPFS was proposed to reduce the weight of the Ethereum blockchain. In this paper, the total size of Ethereum was reduced by uploading the smart contract code in Ethereum and not taking up space due to unnecessary or unused smart contracts, and uploading the smart contract code to IPFS and storing only the IPFS hash. This is a method suitable only for Ethereum that reduces the size of the smart contract code stored in the Ethereum blockchain.

2.2. Consensus Algorithm Based on PBFT

Hyperledger Fabric uses a PBFT-based consensus algorithm [19], and the role of the blockchain node is played by a peer node and orderer node. When a transaction occurs in Hyperledger Fabric, the transaction is sent to the Endorsing Node, and the received node writes its own signature after verifying the validity, and after executing the transaction. After that, the transaction is transferred to the client, and all transactions are sent back to the orderer, and the orderer checks whether the signature of each transaction is valid, creates a block, and delivers the block to peers in the blockchain network. The Peer verifies the validity of the block and adds it to the blockchain if it is valid. In Fusion Chain, the role is not divided into a node that verifies a transaction, and a node that creates a blockchain. The difference is that the client, the creator of the transaction, directly drives the blockchain node. In Hyperledger Fabric, the consensus algorithms of Solo, Kafka, PBFT, and Raft are used.

Cosmos [20] used a consensus algorithm called Tendermint to solve the energy inefficiency of the PoW method of existing blockchain projects, by using an interchain that connects different blockchains. PBFT with a block generator was used. A modified version of the PBFT consensus algorithm was used. After two-thirds or more nodes reach a fast consensus, they create blocks and give finality. To prevent the blockchain from being forked, verifiers use a round robin method to select a verification leader, and the authority is the number of tokens. A blockchain is created in each round, and selected by an ordered list of validators in proportion to their voting rights.

2.3. Privacy in Blockchain

In MedChain: Efficient Healthcare Data Sharing via Blockchain [21], a blockchain is used without using a third-party cloud service to store healthcare data. An efficient healthcare data sharing scheme was proposed, and the roles of users were divided into patient, requester, and healthcare provider. Nodes with each role possess a public key–

private key, and the healthcare provider encrypts the healthcare data collected from the sensor, and stores it in the blockchain, focusing on efficient and safe storage of the data.

In Blockchain Mechanism and Symmetric Encryption in a Wireless Sensor Network [22], the integrity and availability of data collected from IoT devices in a wireless sensor network are maintained using blockchain and cryptographic tools. The plaint-text collected by the IoT sensor uses the AES (advanced encryption standard) encryption algorithm to increase data security.

The Secure Data Sharing Platform Using Blockchain and Interplanetary File System [23] proposed a data sharing scheme using Ethereum's smart contract and IPFS.

2.4. Other Blockchain Platforms

We will briefly describe three representative blockchains. Table 3 compares Fusion Chain with existing representative blockchain platforms.

Table 3. Comparison of roles by blockchain node.

Platform	Scalability	Consensus Algorithm	Node Type	Block Creation	Block Validation	TX Creation/Validation	CPU/GPU Overhead
Bitcoin	O	PoW	Full-Node	O	O	O	High
			Lightweight-Node	X	X	O	Low
Ethereum	O	PoW	Full-Node	O	O	O	High
			Lightweight-Node	X	X	O	Low
Hyperledger Fabric	X	Kafka, Raft, Solo, PBFT (practical Byzantine fault·tolerance).	Peer	X	O	O	Low
			Orderer	O	X	X	Low
* **Fusion Chain**	O	PBFT	-	O	O	O	Low

* Fusion Chain has all functionalities with low CPU/GPU overhead.

2.4.1. Bitcoin

Bitcoin [24] is an online cryptocurrency based on blockchain technology. It was designed to allow individuals to freely conduct financial transactions in a P2P manner, without a central agency, such as a bank. In the Bitcoin network, two types of nodes are available: full node and lightweight-node. The Bitcoin full node [25] synchronizes all blockchain data of Bitcoin blockchain network, and participates in the consensus process as it holds all block data from the genesis block to the present. Bitcoin full nodes have full functionality, including block generation, block verification, transaction generation, and transaction verification. Since Bitcoin adapts PoW as a consensus algorithm, it requires high CPU computation power for mining blocks.

The Bitcoin lightweight-node [26] participates in the blockchain and performs transactions. It requests data from the full node to verify individual transactions. It does not have all the block data like a full node, but only summarizes and holds important data in the Merkle tree in the block header. These nodes can verify and create transactions but cannot participate in the consensus process.

2.4.2. Ethereum

Ethereum [27] is a distributed computing platform for implementing smart contract functions based on blockchain technology. It provides extensibility to operate various applications with smart contracts. The Ethereum full node [28] stores all blockchain data and verifies the new transactions and blocks received, and it participates in the consensus process. Like Bitcoin, it adapts PoW as a basic consensus algorithm, and requires high CPU computation power for mining blocks.

The Ethereum lightweight node [29] does not store all blockchain data, but only stores some information of the blockchain. When storing blocks, the Merkle Patricia tree structure in the block header is used to reduce weight. It has functions related to transactions

that can be performed, but it cannot generate blocks since it cannot participate in the consensus algorithm.

2.4.3. Hyperledger Fabric

Hyperledger Fabric [30] is a platform for developing blockchain solutions and applications. Unlike Bitcoin or Ethereum, hyperledger fabric is a permissioned blockchain, which means that only permissioned nodes can join the private blockchain network. Note that Bitcoin and Ethereum are public blockchains where everyone can join the network. Therefore, instead of allowing access to a private network to specific people and not being paid coins, the advantage is faster network speeds than general public blockchains. Hyperledger Fabric has a peer node and orderer node. The peer node [31] runs in the form of a docker container, verifies the transaction sent by the client, and verifies the block created by the orderer node. It uses PBFT, Kafka, Solo, and Raft as consensus algorithms.

3. Design

In this section, we describe the architecture of the Fusion Chain. It describes the system structure, consensus algorithm, data privacy, and simple implementation notes.

3.1. Overview

Figure 1 shows the whole architecture of the Fusion Chain, and also shows the process of block creation in Fusion Chain. The steps of block creation are as follows.

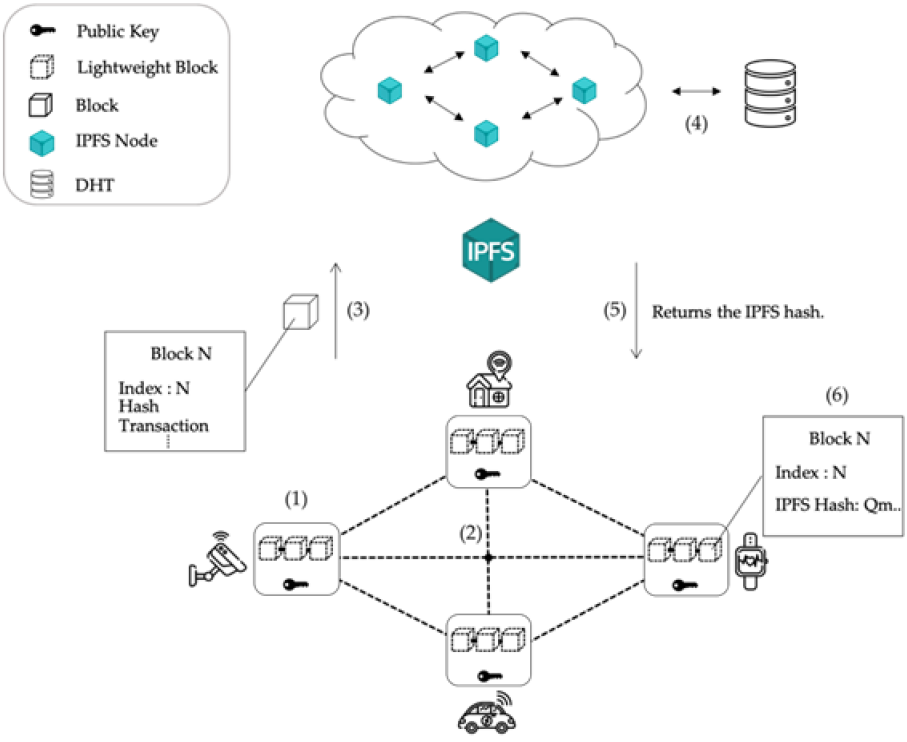

Figure 1. System Architecture.

- Step 1: An IoT device collects data through sensors and encrypts it using its public key. Then it creates a transaction and propagates the transaction to other participants in the network.
- Step 2: All nodes of the Fusion Chain network use PBFT consensus. If it succeeds, a new block is created, which includes the transaction in Step 1.
- Step 3: Uploads the created block to IPFS.
- Step 4: The uploaded block is stored in the distributed hash table (DHT) of the IPFS network.

- Step 5: When blocks are distributed and stored on the IPFS Network, an IPFS hash is returned to Fusion Chain.
- Step 6: The Fusion Chain node creates a lightweight block that stores the index and IPFS hash of the block in the block, and stores it in the blockchain.

Table 4 describes the system layer of Fusion Chain. It has an IoT node, blockchain network and blockchain data store layer.

Table 4. System Layer.

Layer	Features
IoT Node	Through the sensor, data is collected from the surrounding environment and the device performs data encryption with a public key to create a transaction.
Blockchain Network	Blocks are created through transactions. After verifying the block through the PBFT consensus algorithm, it is uploaded to the interplanetary file system (IPFS) network.
Blockchain Data Store	Blocks created in the blockchain network are distributed and stored in the IPFS network, and the IPFS hash is returned.

In the blockchain network, a block is created from the transmitted transaction, and the block is verified through the PBFT consensus algorithm. In IPFS, the verified block is distributed and stored in IPFS, and only the IPFS hash is returned, while only the index and IPFS hash are stored in the block in the blockchain network.

3.2. IoT Node

In the IoT node, data is collected from the surrounding environment through sensors and it is encrypted, and then a transaction is generated and propagated. Fusion Chain uses an asymmetric key algorithm based on PKI algorithms. This algorithm uses two types: a private key that is not shared with anyone, and a public key that is open to everyone. In this algorithm, different keys are used for both encryption and decryption. In this paper, the data generated by the IoT device can be accessed only by authorized users through the private key owned only by the user. The following process describes encrypting and decrypting data generated by IoT devices in Fusion Chain.

(1) The data collected from the IoT is encrypted with the user's public key. A transaction contains this encrypted data, and a block contains this transaction. In Fusion Chain, the block is stored in the blockchain store. In Section 3.4., the detailed process of the blockchain store is described.
(2) When accessing data, the block data is decrypted using a private key owned only by the user. When the data is in the blockchain store, the IoT node uses a hash value of the data to find exact data from the blockchain store.

3.3. Blockchain Network

For the blockchain network, we use PBFT as a consensus algorithm. PBFT was created to improve the speed of the BFT consensus algorithm, which was created due to frequent malicious attacks and errors on the network. PBFT solves the Byzantine general problem while operating in an asynchronous system, and can achieve real-world performance. PBFT needs to obtain more than two-thirds of the agreements of other nodes in order to generate blocks, after verification of all nodes participating in consensus in the network. Since all nodes participating in each verification participate in the case of PoW, there may be a branch in which two different miners finish mining at the same time, and create their own blocks on the blockchain, but in PBFT, since all nodes agree, this type of branch does not occur. Algorithm 1 shows the process of the PBFT consensus algorithm.

Algorithm 1. PBFT Algorithm (f = the number of Byzantine nodes)

1. The client sends a transaction message to the primary node.
2. When the primary node receives the transaction, it executes a procedure called pre-prepare. Then it creates a message, and sends it to all nodes.
3. The node in the network receives the pre-prepare message, and verifies that it is valid. Prepare message is generated, and sent to all nodes only when it is verified.
4. Each node collects pre-prepare messages and prepare messages. When the number of pre-prepare messages is 2f + 1 and prepare messages are 2f or more, the status of prepared certificate is established.
5. When the condition of the prepared certificate is satisfied, a commit message is sent to all nodes. The received node sends a commit message again if valid.
6. Each node collects the commit message and enters the commit certificate status. If there is both a prepared certificate and a commit certificate, it becomes a committed certificate and accepts the transaction requested by the client.

In this paper, PBFT was combined with a block generator and used in a way suitable for Fusion Chain. Unlike Algorithm 1, where the client generating transaction and the node performing verification are separate, in the Fusion Chain network, the node that generated the transaction is the IoT device. Thus, the transaction creator becomes the leader node, and the node generates a block with the encrypted transaction. After that, it sends block verification message to other nodes. The block is verified through a total of three processes: propose, prevote, and commit. Algorithm 2 and Figure 2 describe the PBFT consensus process in Fusion Chain.

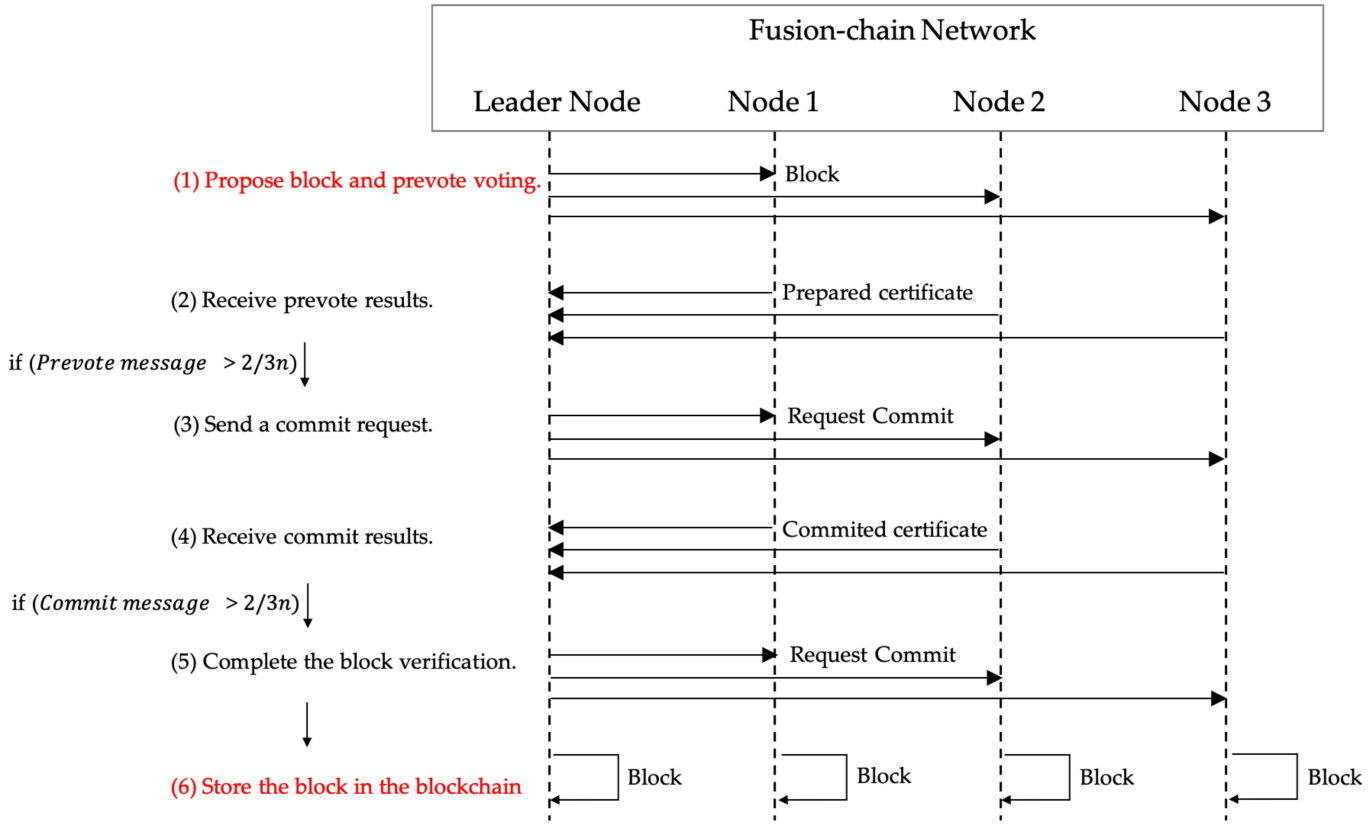

Figure 2. A sequence diagram Fusion Chain with PBFT.

In Algorithm 2, the blockchain node that generated data in Fusion Chain becomes the leader node, creates a block, and requests verification from other nodes. When there are N nodes in the network, and when there are f byzantine failures, N – f messages must

be delivered to operate normally. Furthermore, since there are cases where f byzantine failure nodes send malicious messages, (N-f)-f messages must be more than f messages. (N-f)-f > f $\longrightarrow$ since N > 3f, it is guaranteed that a node consisting of at least 3f + 1 node can withstand f node attacks and errors. Based on the content, a block generator is added to the consensus algorithm of the Fusion Chain to transmit and verify a block instead of a message, and a total of two block verifications are performed: the prevote step, the network preparation step for block verification, and the commit step for block verification. First, the leader node creates and transmits the pre-vote message and block to all nodes. Only when the verification is more than two-thirds of the total nodes, a commit message is requested from all nodes and the block is verified again to two-thirds of the total nodes. If the above is verified, the block is added to the blockchain. This process reduces the possibility of data forgery in the process of adding a block to the blockchain by receiving verification from another node immediately at the node generating the block.

Algorithm 2. PBFT with block generator algorithm for Fusion Chain (n = the number of Fusion Chain nodes)

1. When sensor data is created in an IoT node, the node becomes a leader node, encrypts the data, creates a transaction, creates a block, and requests propose and prevote from other nodes.
2. Each node sends a prevote message to the leader node after verification. The leader node is in prepared certificate status when the number of prevote messages is more than two-thirds of all nodes.
3. The leader node requests a commit message from nodes.
4. Each node sends a commit message to the leader node after verification. Leader node is in the state of commited certificate when the number of commit messages is 2/3n or more of all nodes.
5. If the prepared certificate condition and the commit certificate condition are satisfied, the leader node completes the verification of the block.
6. All nodes store the verified blocks in the blockchain.

3.4. Blockchain Data Store

Fusion Chain leverages IPFS for reducing the storage overhead of blockchain data in the IoT node. IPFS stores files, such as photos, texts, and videos distributed on the internet, and can quickly load distributed data using unique hash values. It is possible to load high-capacity files quickly and efficiently by a unique hash value. Thus, it can be used efficient for file storage. In Fusion Chain, blocks are stored in IPFS instead; the Fusion Chain blockchain only maintains a hash value of the block. A distributed hash table (DHT) [32] allows nodes participating in the IPFS network to manage hash tables individually and store data without a server. Note that DHT is used when searching for a file. This is a method of mapping the file name to a value in the hash table held by each node, without using a central server. DHT can reduce the load on the network, and files on the network can be searched quickly and accurately. IPFS uses a method of finding the name of a file (CID) on DHT through content-addressed data that uses the file itself as an address, and then finding the nodes that have distributed fragments of the file and loading the file. When IPFS distributes files, it converts all files on the network into the Merkle directed acyclic graph(DAG) format. For each node, the CID, the hash of the node contents, is used for Merkle–DAG. Using Merkle–DAG, IPFS can address content, prevent tampering, and prevent duplication. IPFS works by searching and sharing IPFS objects. IPFS objects only store the file contents in binary format when the contents of the original file are smaller than 256 KB, and stores the files in chunks if the contents of the original file are larger than 256 KB.

Figure 3 explains how blocks are stored in IPFS in Fusion Chain. The storage method is as follows.

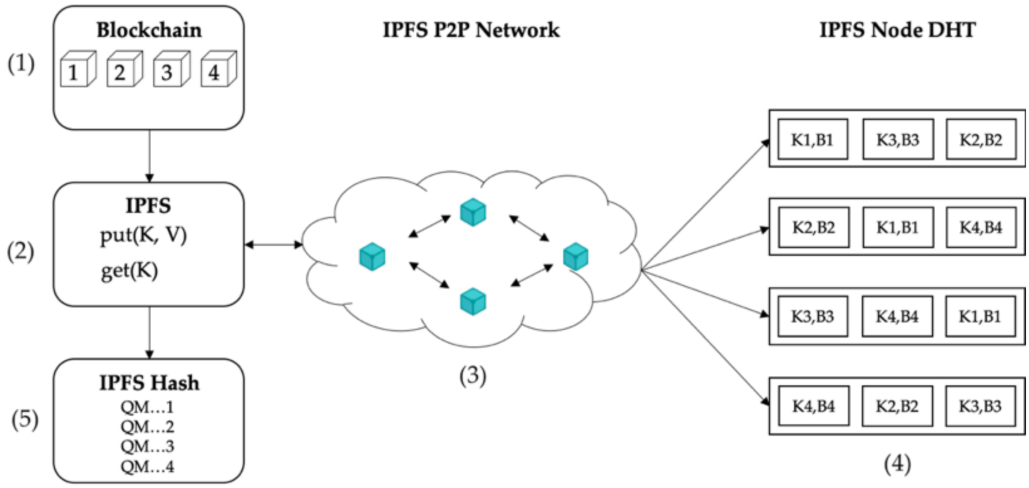

Figure 3. DHT block storage example.

- Step 1: Blocks are created in the blockchain network as a form of file.
- Step 2: The created block is uploaded to the IPFS.
- Step 3: Block files are distributed to nodes on the IPFS P2P network.
- Step 4: The divided files are stored in the DHT of the IPFS nodes. When saved, it is distributed and saved in the form of key-value.
- Step 5: After uploading to IPFS, the IPFS hash is returned, and the hash value is stored in the blockchain.

3.5. Block Structure

Figure 4 depicts the block structure of Fusion Chain. As Fusion Chain is aimed at IoT devices, it stores essential data into a block: Index and IPFS hash. The IPFS stored data contains an index, hash, previousHash, timestamp, and data. Thus, it can maintain a very lightweight structure. In contrast, Ethereum contains almost 20 data fields in a block, which causes high storage overheads.

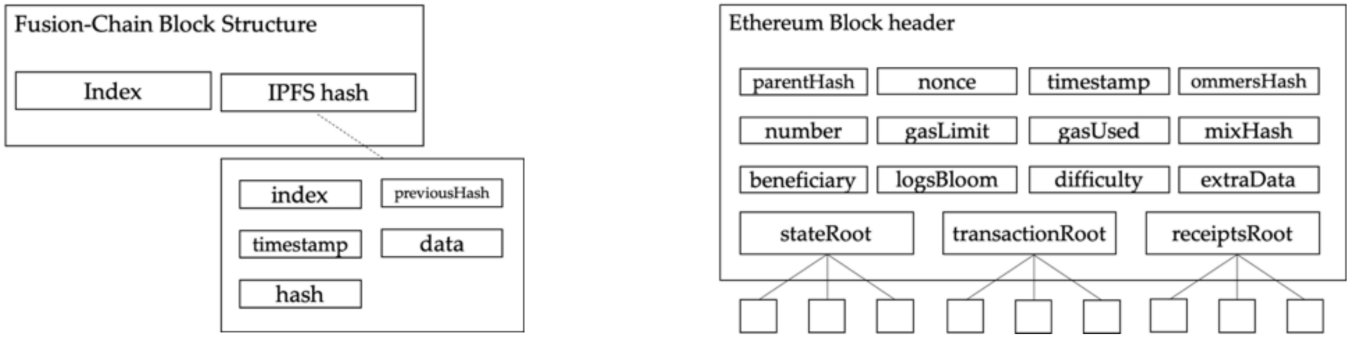

Figure 4. Block structure comparison: Fusion Chain vs Ethereum.

4. Implementation

This section describes the implementation of Fusion Chain. We published the full source code of Fusion Chain in github.com [33]. It was written in Node.js and golang [34] from scratch.

Figure 5 shows the message that the nodes of the blockchain network send and receive during the PBFT consensus process. In the method of applying PBFT proposed in this paper, the node that generated the transaction becomes the leader node, and after generating the block, it propagates to all other nodes, and the prevote process starts. The nodes that receive the block, verify the block, and send the result back to the leader node. If two-thirds or more agree, the commit process begins. Again, the nodes verify

the block. After that, the result is sent to the leader node, and when two-thirds of them agree, the block is added to the blockchain. In this process, the nodes implement request and verification through the message in Figure 5. Through the REQUEST_PREVOTE and REQUEST_COMMIT messages, the Leader node requests prevote and commit verification. Nodes that have received the verification request send verification results through the messages of GET_PREVOTE and GET_COMMIT.

```
var RequestPrevote = (newBlock) => ({
    'type': MessageType.REQUEST_PREVOTE,
    'data': newBlock
});
var RequestCOMMIT = (newBlock) => ({
    'type': MessageType.REQUEST_COMMIT,
    'data': newBlock
});
var sendPreVoteMsg = (newBlock) => ({
    'type': MessageType.GET_PREVOTE,
    'data' : newBlock,
    'count' : 1
});
var sendNotPreVoteMsg = (newBlock) => ({
    'type': MessageType.GET_PREVOTE,
    'data' : newBlock,
    'count' : 0
});

var sendCommitMsg = () => ({
    'type': MessageType.GET_COMMIT,
    'count' : 1

});
var sendNotCommitMsg = () => ({
    'type': MessageType.GET_COMMIT,
    'count' : 0
});
```

Figure 5. PBFT consensus messages.

Figure 6 shows the implementation of the PoW consensus process. Fusion Chain implements PoW for performance comparison. In the consensus process, minor nodes in the blockchain network create a block by obtaining a transaction from the transaction pool, and calculate the hash value through the index, previoushash, timestamp, data, difficulty, nonce, and SHA256 algorithm of the created block. After that, the nonce value is set to 0, and the SHA256 hash value is repeatedly calculated by adding one by one, and the correct answer is repeatedly searched until a hash value with the same number of preceding zeros is found. The number of zeros for the correct answer varies depending on the difficulty level.

```
const calculateHashForBlock = (block: Block): string =>
    calculateHash(block.index, block.previousHash, block.timestamp, block.data, block.difficulty, block.nonce);

const calculateHash = (index: number, previousHash: string, timestamp: number, data: string,
    difficulty: number, nonce: number): string =>
CryptoJS.SHA256(index + previousHash + timestamp + data + difficulty + nonce).toString();

const findBlock = (index: number, previousHash: string, timestamp: number, data: string,
    difficulty: number): Block => {
    let nonce = 0;
    while (true) {
        const hash: string = calculateHash(index, previousHash, timestamp, data, difficulty, nonce);
        if (hashMatchesDifficulty(hash, difficulty)) {
            return new Block(index, hash, previousHash, timestamp, data, difficulty, nonce);
        }
        nonce++;
    }
};
const hashMatchesDifficulty = (hash: string, difficulty: number): boolean => {
    const hashInBinary: string = hexToBinary(hash);
    const requiredPrefix: string = '0'.repeat(difficulty);
    return hashInBinary.startsWith(requiredPrefix);
};
```

Figure 6. Major Functions for PoW consensus algorithm.

5. Experimental Results

5.1. Experimental Environment

The experiment was conducted using three RaspberryPi 4 B (System on Chip: Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz, Memory: 4 GB LPDDR4-3200 SDRAM, OS: Raspbian GNU Linux 10) [35]. The Fusion Chain was implemented through Node.js, and Ethereum through Geth [36]. Node.js used v10.15.3 and Geth version 1.9.6. Geth is an official golang implementation of the Ethereum protocol.

For the experiment, we used four types of common dataset for IoT devices: log, picture, sound, and video. Table 5 contains a detailed dataset description for the experiments. The size of each dataset was determined by referring to the average file size of txt, jpg, wav, and mp4 files [37].

Table 5. Dataset for IoT device.

Data Type	Size	Format
Log	1 KB	txt
Picture	10 KB	jpg
Sound	100 KB	wav
Video	1 MB	mp4

5.2. Blockchain Size Overhead

Fusion Chain maintains IPFS hash values in its blockchain, and data contents are stored in the IPFS. Please note that the size of the IPFS hash value is only 32 Bytes, which means that for a small log file the size difference is 32 times, and for a video file, the difference is 32 k times. Apparently, therefore, the size of this blockchain is much smaller than other existing blockchains like Ethereum or Bitcoin, because Fusion Chain only stores IPFS hash (32 bytes) into its blockchain. For example, if we put a 1 MB video into a blockchain, existing blockchain platforms will store the 1 MB video itself, but Fusion Chain stores only 32 bytes.

However, what if we store 32 bytes of text data in existing blockchain platforms? In order to see the effectiveness of the simple block structure of Fusion Chain, we conducted

more experiments. We generated 1000 blocks in Fusion Chain and Ethereum blockchain, respectively. For fairness, each block contained 32 bytes data, which is the same as the IPFS hash value size. Figure 7 shows the result of the experiment. Surprisingly, the blockchain size of Ethereum was much bigger than Fusion Chain. This difference is caused by the complexity of the Ethereum block structure.

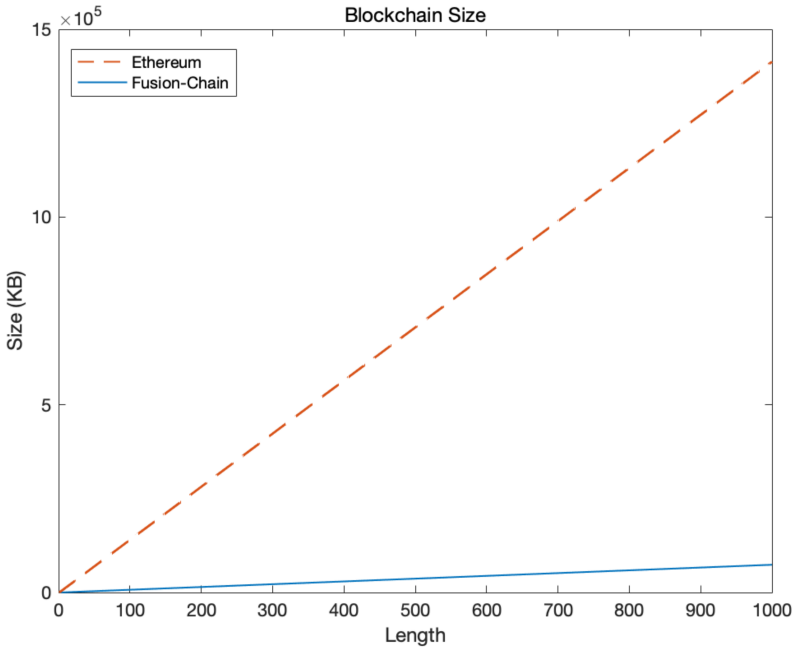

Figure 7. Blockchain Size.

The blockchain size comparison experiment process was as follows: H() is a function that uploads data to IPFS and returns a hash. FusionChainHash of (1) uploads all data except the index of the existing block. ipfsHash uploads and returns only IoT data to be stored in the blockchain, and becomes data in Ethereum transactions. In the block size of (2), FusionChainBlock only has index and FusionChainHash, but EthereumBlock stores BlockHeader and ipfsHash. As a result of this in (3), when a block with a length of 1000 is created, it is found that the blockchain size of Fusion Chain is always smaller than that of Ethereum (4).

$$\text{HashSize} = \begin{cases} \text{FusionChainHash} = \text{H}(\text{ previousHash,timestamp,data,hash}) \\ \text{ipfsHash} = \text{H}(\text{data}) \end{cases} \quad (1)$$

$$\text{BlockSize} = \begin{cases} \text{FusionChainBlockSize} = \text{index} + \text{FusionChainHash} \\ \text{EthereumBlockSize} = \text{EthereumBlockHeader} + \text{ipfsHash} \end{cases} \quad (2)$$

$$\sum_{1}^{1000} \text{FusionChainBlockSize}(\text{data}) < \sum_{1}^{1000} \text{EthereumBlockSize}(\text{data}) \quad (3)$$

All other cases:

$$\text{FusionChain} < \text{Ethereum} \quad (4)$$

5.3. CPU and Memory Overhead

Supporting low CPU and memory overheads is essential requirement for IoT devices. In order to test the CPU and memory overheads, we measured CPU and memory usage during mining operations for Fusion Chain and Ethereum. Originally Fusion Chain's default consensus algorithm was PBFT. In addition, we implemented the PoW consensus algorithm in Fusion Chain for the sake of a fair comparison with Ethereum, as it supports PoW only.

Figure 8 depicts CPU usage when Fusion Chain and Ethereum ran consensus algorithms. The PBFT consensus algorithm showed very low CPU usage (about 6%) compared to PoW. The PBFT algorithm makes consensus using network communication among the participating nodes. Therefore, CPU computation is hardly performed. In the case of PoW, however, since mining nodes create blocks and perform hash calculations with the index, previoushash, timestamp, data, difficulty, and nonce to find a specific hash value, it consumes lots of CPU computation power. In the case the PoW, Ethereum and Fusion Chain used 110% and 100% of the CPU, respectively. Note that, Raspberry PI 4 B has a quad core CPU, so a maximum 400% of CPU usage is available. As a result, the Fusion Chain consensus algorithm is almost 16 times lighter than Ethereum's consensus algorithm.

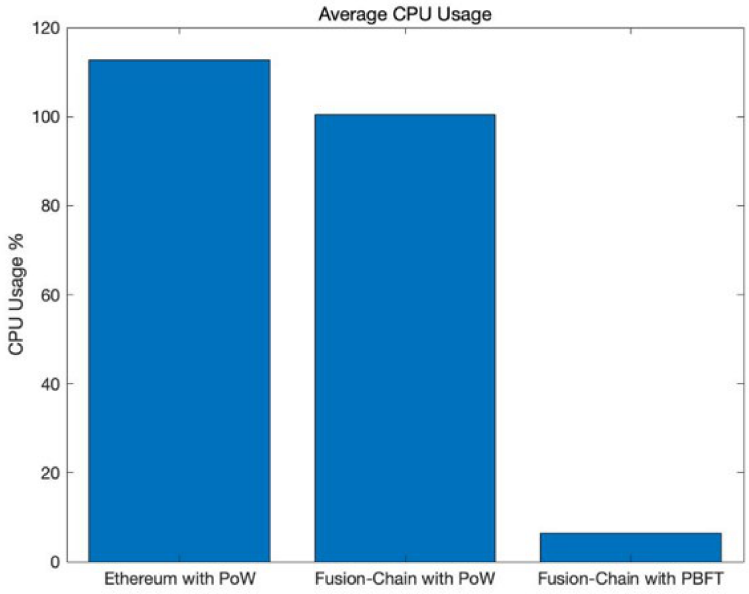

Figure 8. CPU usage comparison.

Figure 9 depicts memory usage when Fusion Chain and Ethereum run the consensus algorithm. Compared to the PBFT consensus algorithm, PoW must maintain a transaction pool to store transactions in memory. For this reason, PBFT is 49.6 MB, and PoW is 64.4 MB on average, which uses about 30% more memory than PBFT. In the case of Ethereum, an average of 2.1 GB of memory is used during the mining process.

5.4. IPFS Latency of Upload/Cat

Fusion Chain needs to access IPFS when it needs to access its block data. In this experiment, we measured IPFS upload and download latency. Table 6 shows the experimental results of measuring the upload latency time. Internally, Fusion Chain uses the IPFS command below to upload data to IPFS.

$$\text{ipfs add } \langle \text{path} \rangle \ldots \; - \; \text{Add a file or directory to ipfs}$$

Table 6. IPFS Upload Latency.

Data Type	Maximum	Minimum	Median
Log	104.97 ms	85.801 ms	91.632 ms
Picture	115.81 ms	84.635 ms	93.43 ms
Sound	123.7 ms	87.733 ms	97.877 ms
Video	155.98 ms	117.76 ms	127.91 ms

Figure 10 describes the latency time that occurs when uploading the data set in Table 5 to IPFS. The latency time for uploading log, picture, sound, and video data to IPFS was

measured. Table 6 shows the upload time in a table. The log was the smallest, and the latency time of other data did not change significantly depending on the size, so the result is suitable for large-sized data.

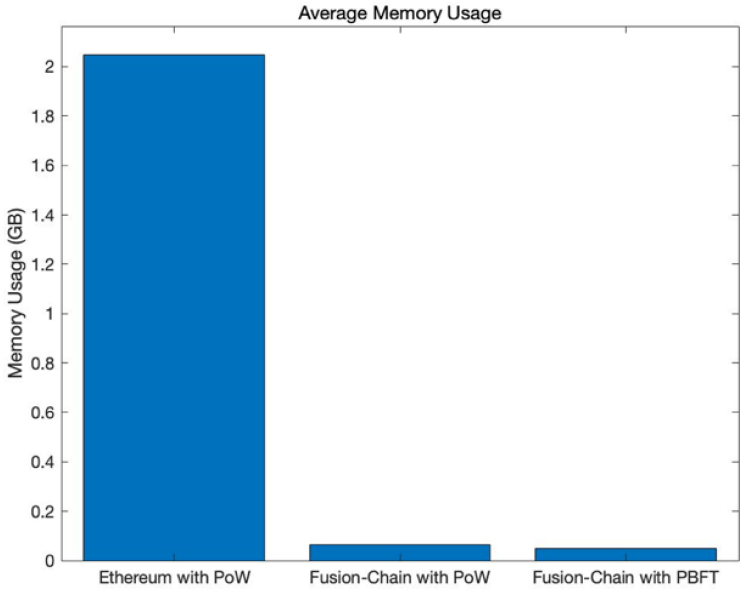

Figure 9. Memory usage comparison.

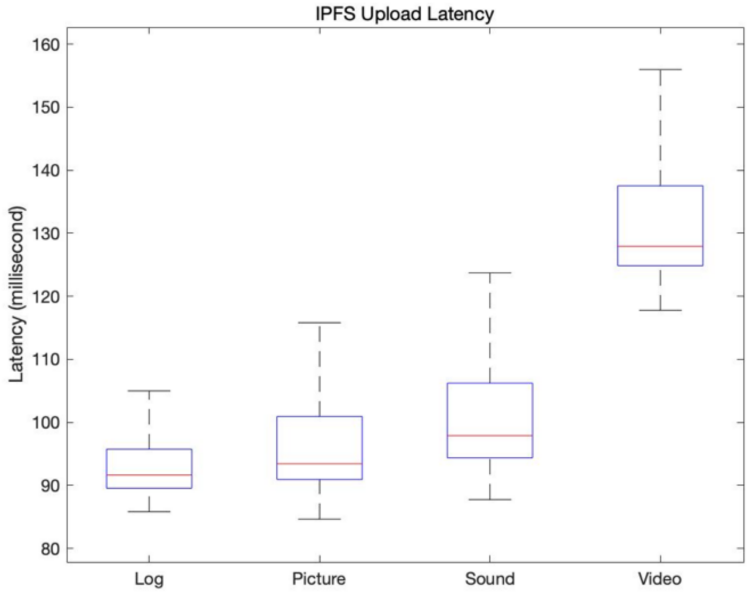

Figure 10. IPFS Upload Latency.

In the process of downloading the data set of Table 5 from IPFS, Table 7 is the result of measuring the latency time when downloading the IPFS hash of the file uploaded to IPFS. Internally, Fusion Chain uses the IPFS command below to download data from IPFS. Note that the cat command downloads data to memory rather than to disk.

$$\text{ipfs cat } \langle \text{ipfs} - \text{path} \rangle \ldots \; - \; \text{Show IPFS object data.}$$

Figure 11 depicts latency time when downloading the data set from Table 5. There are two types of file download methods of IPFS: disk download through get command, and memory download through cat command.

Table 7. IPFS Cat Latency.

Data Type	Maximum	Minimum	Median
Log	87.968 ms	75.715 ms	80.097 ms
Picture	97.287 ms	75.448 ms	80.806 ms
Sound	94.959 ms	79.263 ms	84.905 ms
Video	121.76 ms	88.139 ms	98.572 ms

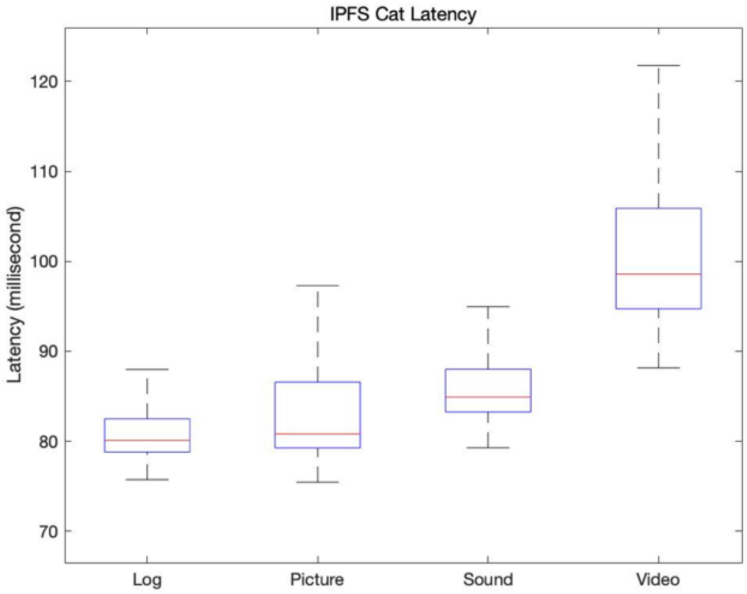

Figure 11. IPFS Cat Latency.

5.5. PKI Latency

Figure 12 shows the result of measuring the latency time when data is encrypted and decrypted with an asymmetric key. In the case of encryption with a public key, there was a latency of 49.7734 ms on average, and in case of decryption, there was a latency of 1.0173 ms. The latency time of encryption was more than that of decryption, and the latency time when applied had a small effect on performance.

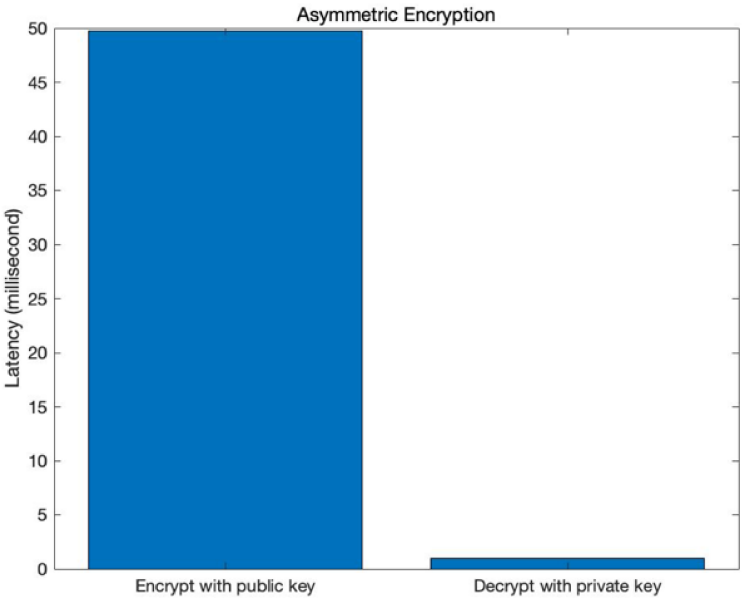

Figure 12. Asymmetric Encryption Latency.

6. Conclusions

In this paper, a lightweight blockchain named Fusion Chain was proposed for IoT devices. A solution was proposed to solve the specific problems (1), (2), and (3). As a result of the experiment (1), the size of the blockchain was reduced from a maximum of 1010.12% to a minimum of 270.12%. The computational power problem of (2) proved that the PBFT, which does little CPU computation, uses only 6.3102% of the CPU, a small amount. The PKI of (3) guarantees the privacy of the data, and there was almost no latency time, so it was shown that there is no problem in performance. Through these results, it was shown that Fusion Chain can operate properly in IoT devices, and contribute to the security of IoT devices. As a result, it is possible to solve the data reliability and the security vulnerability of IoT applications, caused by Mirai botnet and DDoS attacks, by using blockchain technology for lightweight IoT devices. The lightweight IoT devices ensure the reliability and integrity of data without a centralized server. Currently we are conducting research on a lightweight blockchain that maintains high transaction per second (TPS) by implementing an inter-chain structured blockchain that divides IoT devices into groups using Fusion Chain.

Author Contributions: Conceptualization, D.N.; methodology, D.N.; Programming, D.N.; writing-original draft preparation, D.N.; supervision and review, S.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Bisa Research Grant of Keimyung University in 2018.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Internet of Things (IoT)—Statistics & Facts. Available online: https://www.statista.com/topics/2637/internet-of-things (accessed on 15 June 2020).
2. Blockchain IoT Market by Offering (Hardware, Software, and Infrastructure Provider), Application (Smart Contract, Data Security, Data Sharing/Communication, and Asset Tracking & Management), End User, and Geography—Global Forecast to 2024. Available online: https://www.marketsandmarkets.com/Market-Reports/blockchain-iot-market-168941858.html (accessed on 26 March 2019).
3. Data Volume of Internet of Things (IoT) Connections Worldwide in 2019 and 2025. Available online: https://www.idc.com/getdoc.jsp?containerId=prUS45213219 (accessed on 18 June 2019).
4. IoT Security: Needed Now More than Ever. Available online: https://cisomag.eccouncil.org/iot-security-needed-now-more-than-ever/ (accessed on 18 June 2019).
5. The Mirai Botnet Attack and Revenge of the Internet of Things. Available online: http://varonis.com/blog/the-mirai-botnet-attack-and-revenge-of-the-internet-of-things/ (accessed on 19 June 2020).
6. Hacker Leaks Passwords for More than 500,000 Servers, Routers, and IoT Devices. Available online: https://www.zdnet.com/article/hacker-leaks-passwords-for-more-than-500000-servers-routers-and-iot-devices/ (accessed on 19 January 2020).
7. Qaisar, S.; Abdul, B. DDoS Botnet Prevention using Blockchain in Software Defined Internet of Things. In Proceedings of the 2019 16th International Bhurban Conference on Applied Sciences & Technology (IBCAST), Islamabad, Pakistan, 8–12 January 2019.
8. Bitcoin Core. Available online: https://bitcoin.org/en/bitcoin-core/ (accessed on 18 July 2017).
9. Ray, P.P. A survey on Internet of Things architectures. *J. King Saud Univ. Comput. Inform. Sci.* **2018**, *30*, 291–319.
10. Ethereum Archive Nodes Now Take up 4 Terabytes of Space. Available online: https://decrypt.co/24779/ethereum-archive-nodes-now-take-up-4-terabytes-of-space (accessed on 8 April 2020).
11. FTC and TrendNet Settle Claim over Hacked Security Cameras. Available online: https://www.cnet.com/news/ftc-and-trendnet-settle-claim-over-hacked-security-cameras/ (accessed on 4 September 2013).
12. IPFS. Available online: https://ipfs.io (accessed on 5 April 2018).
13. Miguel, C.; Barbara, L. Practical Byzantine Fault Tolerance. In Proceedings of the Third Symposium on Operating Systems Design and Implementation, New Orleans, LA, USA, 22–25 February 1999.
14. Ray, H. PKI and Digital Certification Infrastructure. In Proceedings of the Ninth IEEE International Conference on Networks, ICON 2001, Bangkok, Thailand, 10–11 October 2001.

15. Abdur, R.S.; Corey, S.; Rain, K.; Pissinou, N. Sensor-Chain: A Lightweight Scalable Blockchain Framework for Internet of Things. In Proceedings of the Conference: IEEE international conference on Internet of Things, Atlanta, GA, USA, 15 May 2019.
16. Foundation IOTA. Available online: https://www.iota.org (accessed on 13 November 2018).
17. IOTA Attacked on January 19th 2018/IOTA Technology Is Secure. Available online: https://iota-news.com/iota-attack/ (accessed on 17 June 2019).
18. Norvill, R.; Pontiveros, B.B.F.; State, R.; Cullen, A. IPFS for Reduction of Chain Size in Ethereum. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1121–1128.
19. Nguyen, T.S.L.; Jourjon, G.; Potop-Butucaru, M.; Thai, K.L. Impact of network delays on Hyperledger Fabric. In Proceedings of the IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Paris, France, 29 April–2 May 2019; pp. 222–227.
20. Cosmos-Whitepaper. Available online: https://cosmos.network/cosmos-whitepaper.pdf (accessed on 20 June 2020).
21. Shen, B.; Guo, J.; Yang, Y. MedChain: Efficient Healthcare Data Sharing via Blockchain. Proceedings of the Blockchain Mechanism and Symmetric Encryption in a Wireless Sensor Network. *Appl. Sci.* **2019**, *9*, 1207.
22. Guerrero-Sanchez, A.E.; Rivas-Araiza, E.A.; Gonzalez-Cordoba, J.L.; Toledano-Ayala, M.; Takacs, A. Blockchain Mechanism and Symmetric Encryption in A Wireless Sensor Network. *Sensors* **2020**, *20*, 2798. [CrossRef] [PubMed]
23. Naz, M.; Al-Zahrani, F.A.; Khalid, R.; Javaid, N.; Qamar, A.M.; Afzal, M.K.; Shafiq, M. A Secure Data Sharing Platform Using Blockchain and Interplanetary File System. *Sustainability* **2019**, *11*, 7054. [CrossRef]
24. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. bitcoin org. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 24 February 2020).
25. Bitcoin Full Node. Available online: https://en.bitcoin.it/wiki/Full_node (accessed on 23 June 2020).
26. Bitcoin Lightweight Node. Available online: https://en.bitcoin.it/wiki/Lightweight_node (accessed on 15 January 2018).
27. Wood, G. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Proj. Yellow Pap.* **2014**, *2014*, 1–32.
28. EthHub. Available online: https://docs.ethhub.io/using-ethereum/running-an-ethereum-node/ (accessed on 8 October 2019).
29. Ethereum Light-Client-Protocol. Available online: https://eth.wiki/en/concepts/light-client-protocol (accessed on 6 November 2020).
30. Cachin, C. Architecture of the hyperledger blockchain fabric. *Workshop Distrib. Cryptocurrencies Consens. Ledgers* **2016**, *310*, 4.
31. Hyperledger Fabric Peers. Available online: https://hyperledger-fabric.readthedocs.io/en/release-2.2/peers/peers.html (accessed on 18 November 2020).
32. DHT. Available online: https://docs.ipfs.io/concepts/dht/ (accessed on 16 December 2020).
33. Fusion Chain. Available online: https://github.com/sslab-kmu/Fusion-Chain (accessed on 14 September 2020).
34. Go Programming Language. Available online: https://golang.org (accessed on 10 November 2009).
35. Raspberry Pi4. Available online: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/ (accessed on 24 June 2019).
36. Geth. Available online: https://geth.ethereum.org (accessed on 19 August 2015).
37. Average File Sizes. Available online: https://blog.online-convert.com/average-file-sizes/ (accessed on 25 February 2015).