applied sciences

Article

A Lightweight Hash-Based Blockchain Architecture for Industrial IoT

Byoungjin Seok, Jinseong Park and Jong Hyuk Park *

Department of Computer Science and Engineering, Seoul National University of Science and Technology, Gongneung-ro, Nowon-gu, Seoul 01811, Korea; sbj7534@seoultech.ac.kr (B.S.); jseongpark157@seoultech.ac.kr (J.P.)
* Correspondence: jhpark1@seoultech.ac.kr; Tel.: +82-02-970-6702

Received: 27 May 2019; Accepted: 4 September 2019; Published: 7 September 2019

Abstract: Blockchain is a technology that can ensure data integrity in a distributed network, and it is actively applied in various fields. Recently, blockchain is gaining attention due to combining with the Internet of Things (IoT) technology in the industrial field. Moreover, many researchers have proposed the Industrial IoT (IIoT) architecture with blockchain for data integrity and efficient management. The IIoT network consists of many heterogeneous devices (e.g., sensors, actuators, and programmable logic controllers (PLC)) with resources-constrained, and the availability of the network must be preferentially considered. Therefore, applying the existed blockchain technology is still challenging. There are some results about the technique of constructing blockchain lightly to solve this challenge. However, in these results, the analysis in perspective of cryptographic performance (area, throughput, and power consumption) has not been considered sufficiently, or only focused on the architecture of the blockchain network. The blockchain technology is based on cryptographic techniques, and the main part is a cryptographic hash function. Therefore, if we construct the blockchain-based IIoT architecture, we have to consider the performance of the hash function. Many lightweight hash functions have been proposed recently for the resource-constrained environment, and it can also be used to the blockchain. Therefore, in this paper, we analyze the considerations of lightweight blockchain for IIoT. Also, we conduct an analysis of lightweight hash for blockchain, and propose a new lightweight hash-based blockchain architecture that can change the hash algorithm used for mining adjust to network traffic.

Keywords: industrial IoT; blockchain; lightweight hash; security; resource-constrained devices

1. Introduction

Blockchain is a technology that can ensure data integrity by consisting of the immutable ledger in distributed network [1,2]. The data blocks are connected with the hash value of each previous block, and the ledger, including all of the information about the transactions, which is stored in each node in the distributed network. The distributed ledger is updated by making a consensus using defined consensus protocol such as Proof of Work (PoW), Proof of Stake (PoS), Proof of Property (PoP), it helps entire sharing of the same data blocks for all nodes of the network [3]. These features are the key features of the data integrity in the network [4]. Besides, blockchain can be a potential solution for IoT, such as expanding of address space, identify things, governance, data authentication, authorization, and privacy [5]. For these reasons, blockchain-based IoT networks are gaining attention. As the IoT technology extends its area to the various field, many researchers are trying to apply the blockchain-based IoT technology to their interest fields. Notably, in the industrial field, IIoT brings to advantages (e.g., operation automatically, cost reduction, boost productivity, etc.) [6,7]. Additionally, if the blockchain combines IIoT, it can also serve useful services such as on-demand manufacturing, smart diagnostics, machine maintenance, traceability, supply chain tracking, etc. [8]. As a result, many researchers have shown interest in blockchain-based IIoT. However, the IoT network consists of many heterogeneous devices with resource-constrained environment, and the availability issue on blockchain-based IoT network remains an open challenge [9].

In the industrial field, the availability issue is more critical compared to the general blockchain-based IoT network. The use of IIoT networks in the industrial domain, such as Smart Factory, Industrial Control Systems (ICS), and Supervisory Control And Data Acquisition (SCADA), can cause a disaster or loss of money if they do not initiate their functions at the right time [10,11]. It means that the availability of the industrial network is the primary concern. To solve this challenge, there are some existing results for constructing the blockchain network more lightly. In these results, either a new architecture of the blockchain-based IoT network was designed, or the block structure and consensus protocol of blockchain was modified. However, the selection of cryptographic algorithms have not been considered deeply in the results. In the blockchain network, the mining process has mainly a performance issue because it requires high computational power for searching a hash value that satisfies the condition of validating. However, IoT devices that usually consist of the resource-constrained devices are compactly designed to perform their specific purpose. Therefore, they do not have sufficient memory, power, for calculating or area for implementing a hardware module, because the device can perform a specific purpose. For this reason, the hash functions which are generally used may cannot be applied or show improper performance. This limitation can make a latency during the mining process in the blockchain. If it is delayed, the updating block process is also delayed. This means that the performance of the hash function can affect the period of updating block, thereby it cannot serve the sufficient availability of the network. In reality, even though the target updating time is ~10 min in the case of Bitcoin, there are some cases that the updating block time is ~40 min. If the block updating process is delayed in blockchain-based IIoT, it can result in a severe impact. Therefore, the selection of hash algorithms has to be conducted very carefully according to available resources. Recently, many lightweight hash functions have been proposed for use on resource-constrained devices, and it can be a proper solution to select the hash function in the industrial field. Following our survey results, some lightweight hash functions show better performance than SHA-2.

In this paper, we study blockchain IIoT architecture to improve the availability using lightweight hash functions. First, we survey the existing results of blockchain-based IoT and analyze the considerations for blockchain-based IIoT. Then, we conducted surveys for lightweight hash functions and select some hash functions through the performance analysis. Finally, we propose a lightweight hash-based blockchain architecture for IIoT. This architecture includes the blockchain network between the field device layer and control layer, and thereby it can ensure the data integrity of the collected data and control command. Our main contributions can be summarized below.

- We survey existing blockchain-based IoT solutions and summarize their primary considerations in the perspective of modifying the consensus protocol, constructing an architecture, using lightweight cryptography for performance.
- We analyze the considerations for blockchain-based IIoT in perspective of improving availability.
- We survey the lightweight hash functions and conduct a performance analysis. Finally, we propose a lightweight hash-based blockchain architecture for IIoT.

The remainder of this paper is organized as follows. In Section 2, we introduce related work about blockchain-based IoT, hash functions of existing cryptocurrency and we also explain considerations (computational resources, latency, and scalability) for the availability of blockchain-based IIoT. In Section 3, we propose blockchain-based IIoT architecture operating between field layer and control layer. Also, we explain the features of our blockchain architecture and interaction process. In Section 4, we explain how our proposed architecture satisfies considerations for availability. Last, in Section 5, we conclude our work and present future work directions.

2. Related Work

2.1. Blockchain Based IoT

Blockchain is emerging as a solution to solve challenges such as security and privacy in the IoT network by storing blocks to distribute the devices. For applying the blockchain to resource-constrained IoT devices, many researchers proposed some blockchain-based IoT techniques that are designed to operate lightly. Fernández-Caramés et al. [12] surveyed Blockchain-based IoT (BIoT). They described the basics of blockchain and how to impact BIoT application to traditional IoT. Lastly, they suggested some recommendations for BIoT researchers. Novo [13] proposed a decentralized access control architecture based on blockchain for managing a huge amount of IoT devices. In this architecture, the IoT devices do not participate in the blockchain network due to limited resources. Instead, there is some management hub to transfer registration data to the blockchain network. Dorri et al. [14,15] proposed a blockchain-based IoT architecture for security and privacy, exemplifying the smart home, which is one of IoT's applications. They composed three layers consisting of smart home, cloud storage, overlay in their proposed architecture and eliminate the PoW and the concept of coins for using blockchain lightly. Similarly, Biswas et al. [16] also a proposed security framework to deal with some security threats using a private blockchain to eliminate consensus for performance, scalability, and security of the smart city. However, instead of the way which eliminates consensus, Sukhwani et al. [17] proposed the consensus, which improves Practical Byzantine Fault Tolerance (PBFT) using Stochastic Reward Nets (SRN). Boudguiga et al. [18] proposed IoT software update architecture based on blockchain to ensure the availability and accountability of updates. They focused on the availability of an update that it can always be available for target devices, and Li et al. [19] a proposed the blockchain-based decentralized data storage for IoT. They composed blockchain by adding some edge node. The edge node performs cryptographic computations and collects data from IoT devices. Also, they used certificateless cryptography to make the authentication process lighter in their blockchain network. M Samaniego et al. [20,21] proposed a software-defined IoT architecture with blockchain. In this work, they applied blockchain technology from the perspective of blockchain as a service (BaaS). Moreover, the blockchain was used as a virtual resource to push code. Sharma et al. [22] proposed a DistBlockNet blockchain architecture of IoT Networks. The DistBlockNet consists of SDN-based network and blockchain. Also, the proposed architecture is designed for security, scalability, and efficiency. Recently, Liu et al. [23] proposed lightweight blockchain system for applying the IoT environment. They suggested consensus mechanism called "Synergistic Multiple Proof of Work", and also suggested blockchain architecture to apply IIoT environments. Moreover, they show computational cost reduce through experiments.

All of these existing results consider resource-constrained devices when applying the blockchain. However, blockchain has various components related to performance, and researchers conducted their research with different perspectives. We conduct a categorization of an existing blockchain-based IoT techniques (Table 1). Following our taxonomy, most of them do not consider lightweight cryptography. The lightweight hash function can be a good opportunity to construct a lightweight blockchain for IoT, especially in the industrial field, because the blockchain is based on hashchain.

Table 1. Taxonomy of strategies of existing blockchain-based IoT.

Ref.	Description	Strategy for Lightweight		
		Architecture	Consensus	LW Cryptography
Novo [13]	Decentralized access control architecture	✓		
Dorri et al. [14,15]	Three-layer blockchain architecture for smart home	✓	✓	

Table 1. *Cont.*

Ref.	Description	Strategy for Lightweight		
		Architecture	Consensus	LW Cryptography
Biswas et al. [16]	Using private blockchain and eliminate consensus		✓	
Sukhwani et al. [17]	Using SRN to improve PBFT consensus		✓	
Boudguiga et al. [18]	IoT software update architecture	✓		
Li et al. [19]	Decentralized data storage for IoT	✓		✓
M Samaniego et al. [20,21]	software-defined IoT architecture with blockchain	✓		
Sharma, P.K. et al. [22]	SDN based network using blockchain	✓		
Liu et al. [23]	Synergistic Multiple Proof and lightweight data structure	✓	✓	

2.2. Hash Functions on Cryptocurrency

Cryptocurrency is one of the typical applications of blockchain. There are many cryptocurrencies in the real world, and each cryptocurrency has different features; recently, a cryptocurrency changed the hash function or modified the consensus protocol, improving its performance. Bitcoin [1] is the most popular and first-established cryptocurrency. Bitcoin uses PoW that is the first consensus protocol for cryptocurrency. PoW protocol is a costly computer computation involving hashing (SHA-256, scrypt, etc.), Merkle Tree, and peer-2-peer networking for creating, broadcasting, and validating a block on the blockchain network. The bitcoin uses the SHA-256 function; other cryptocurrencies using SHA-256 function are Bitcoin Cash, Counterparty, MazaCoin, etc. An alternative hash function used in many other cryptocurrencies is scrypt [24]. However, it shows the lower hash rate than SHA-256 function. Cryptocurrencies that uses Scrypt are Bitconnect, Bitcoin Gold, Litecoin [25], etc. There is another aggressively grown platform called Ethereum [26] which also uses blockchain technology to facilitate smart contracts and uses PoS protocol. The PoS is more energy efficient and secure protocol compared to PoW counterparts. Ethereum-based coins use the Ethash function, which uses a Keccak hash function [27] that is eventually standardized to SHA-3. SHA-3 is the latest hash function based on sponge function. It reduced the risk of all standard cryptographic algorithms being broken simultaneously. Moreover, blockchain-based coins use various hash functions, and are summarized in Table 2.

Table 2. Top 10 hash functions for blockchain-based coins [28].

Hash Function	Example Coins	Number of Coins
Scrypt	Auroracoin, Bitconnect, Bitcoin Gold, Coinye, Dogecoin, Gridcoin, Litecoin, PotCoin	261
X11	Dash, Petro	106
SHA-256	Bitcoin, Bitcoin Cash, Counterparty, MazaCoin, Namecoin, NeuCoin, Nxt, Peercoin, Titcoin	99
Quark	PIVX, LockChain, DimeCoin	35
CryptoNight	ByteCoin, Electroneum, Fantom	30

Table 2. *Cont.*

Hash Function	Example Coins	Number of Coins
X13	Bitcoin Diamond, Straits, Navcoin	27
Ethash	Ethereum, Ethereum Classic	22
NeoScrypt	Red Pulse Pheonix	22
Equihash	Zcash, Zcoin	13
Keccak	SmartCash, Nexus, Maxcoin	10

2.3. Considerations for Blockchain Based IIoT

IIoT comprises many heterogeneous devices with limited resources. If we apply the existing blockchain technology, then it can affect the availability of the network. Therefore, when we consider the blockchain solution for IIoT, we have to consider this restriction. To improve the availability of blockchain-based IIoT, we have to consider below:

1. Computational resources:The consensus algorithms employed in the mining process of the blockchain (PoW) require significant computational resources that are far beyond the capabilities of most IoT devices. Resource requirements [29] depend on the particular type of consensus protocol in the blockchain network. Typically, solutions tend to delegate these tasks to gateways, or to any other unconstrained devices capable of providing this functionality. Although there are initiatives to incorporate blockchain full nodes into IoT devices [30]; mining is still a key challenge for IoT due to its limitations. To solve this challenge, we have to consider the computational burden of consensus algorithms. That is, we have to consider whether the devices on the network can adequately serve their functionality during the mining process. Therefore, we have to apply or design a lightweight consensus algorithm, especially in the industrial field.
2. Latency: There is a significant latency associated with ensuring that transaction is confirmed by nodes which participating in the blockchain network. For example, in Bitcoin, it can take up to 30 min for a transaction to be confirmed. Most IoT applications have latency requirements, e.g., a service provider requesting data from a smart home sensor should not have to wait for several min. However, the latency requirements are stricter in the industrial field because IIoT can be used with huge manufacturing control systems or national infrastructures. For ensuring the latency, we have to use fast cryptographic functions and consensus algorithms for fast operation of the blockchain network.
3. Scalability: In a typical blockchain implementation, the target block for updating is broadcast and verified by all nodes. It leads to significant scalability issues since the broadcast traffic and processing overheads would increase quadratically with the number of nodes in the network. Especially in the case of the industrial field, the devices are placed in a wide location. The associated overheads are intractable as most IoT devices have limited bandwidth connections (e.g., Low-Power Wide-Area Networks such as LoRa network which has a bandwidth under 1 GHz.) and processing capabilities. Therefore, the blockchain network has designed to broadcast efficiently.

3. Proposed Lightweight Hash Based Blockchain Architecture for IIoT

3.1. Overview of Proposed Lightweight Hash Based Blockchain Architecture

In this section, we propose a lightweight, hash-based blockchain architecture for IIoT. In the previous section, we mentioned that the selection of hash function can affect the performance of the blockchain network due to the computational burden for block mining, which means that the performance of the hash function can affect the availability of the blockchain. However, IIoT devices have limited resources (low power, small area, and small memory). We selected some lightweight hash functions which can be implemented using small area and perform with low-power, small memory. The proposed blockchain network consists of "Cell node" and "Storage node", and it operates between the field layer and control layer. We referred to the Purdue model to design the proposed architecture. Purdue model (ISA-95) is a standardized reference model proposed by Theodore J. Williams [31] for Industrial Control System (ICS). Field layer of our proposed architecture correspond with level 0 and level 1 in Purdue model. And control layer of our proposed architecture correspond with level 2 in Purdue model. For covering many heterogeneous devices in a broad area, we separate the field to a small areas, which is mentioned as "Cell", and all IIoT devices connected to cell node, which is nearly located. And the cell node makes a block from data gathered by connected devices and broadcast to other nodes in blockchain for block validation after block mining. After the block validation process, all node participating in block validation send the return message to the storage node for notice validating results and then block update is processed. The storage node is responsible for managing block update and ledger management. In the block update process, the storage node appends the validated block. All of the processed transaction can be checked from the distributed ledger in the storage node. Figure 1 shows our proposed architecture, and brief explanations of the cell node and storage node are as follows, and we discuss more detailed description and process of each function in Section 3.3.

- Cell node: The cell node is a miner node in blockchain, and it consists of "LW Hash Manager, Device Manager, Traffic Analyzer, Mining Pool, Block Mining Manager, and Block Validation Manager". The LW hash manager contains lightweight hash functions used in blockchain. The device manager is responsible for device registration required for participation in the cell area. The gathered data is stored in the mining pool as transactions that have yet to be processed. The traffic analyzer investigates the number of transactions. The calculated amount used to select a hash function from the lightweight hash list for block mining according to the predefined threshold. The hash list includes three hash functions—QUARK, PHOTON, and SPONGENT; please see the detailed description of the lightweight hash list in Section 3.2.1. The block mining manager operates block construction and searches the hash value, which satisfies the target difficulty. Last, the block validation manager operates the block validation process when other nodes request a block validation. Using these components, the cell node performs three functions (device management, block mining management, and block validation management) in the blockchain network. Please find the explanation and related component of each function in Table 3.
- Storage node: The storage node is a full node in blockchain, and it consists of LW Hash Manager, Block Validation Manager, and Shared Ledger Manager. The LW hash manager and the block validation manager have the same functions as the cell node. In contrast to the cell node, the storage node contains the shared ledger manager, and it manages the ledger, which contains all of the processed transactions. Also, the storage node does not operate block mining, and it performs just two functions (block validation management, block update management). After the block validation process, the shared ledger manager appends the validated block to the shared ledger. Please find the explanation and related component of each function in Table 4.

Table 3. Functions and related component of the cell node.

Function	Description	Related Component
Device management	Device registration for participating in the cell area	Device Manager
Block mining management	Choosing the hash function used for block mining, Block construction and perform mining process	LW Hash Manager, Traffic Analyzer, Mining Pool, Block Mining Manager
Block validation management	Validating the block requested from the other node and return the result of the validation process	Block Validation Manager

Table 4. Functions and related component of the storage node.

Function	Description	Related Component
Block validation management	Validating the block requested from the other node and return the result of the validation process	Block Validation Manager
Block update management	Appending the validated block to the stored ledger	Shared ledger manager

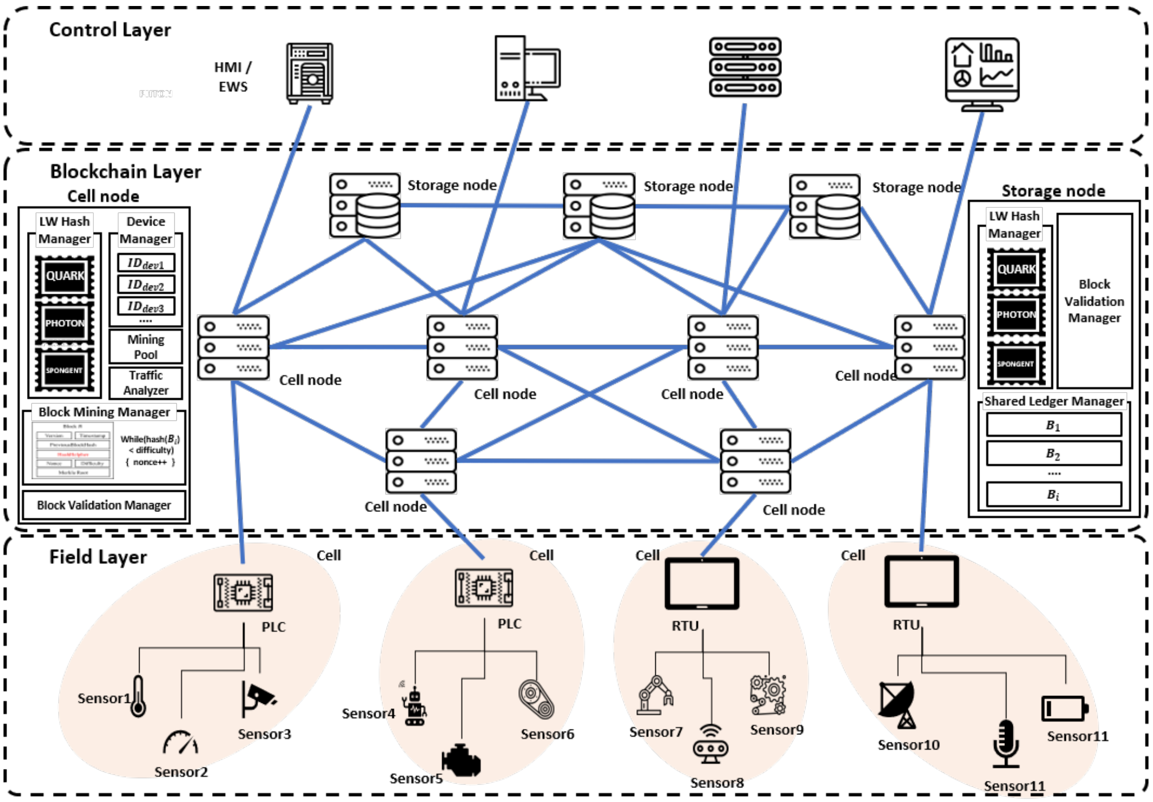

Figure 1. Overview of proposed lightweight hash-based blockchain architecture.

3.2. Key Features of Proposed Blockchain

3.2.1. Selection of Hash Functions: Lightweight Hash List

We created a lightweight hash list in the LW hash manager, containing lightweight hash functions that are used for block mining. When we constructed the hash list, we aimed to select the proper hash algorithm, which can serve security cryptographically and can be implemented using a small area. Moreover, we selected some lightweight hash functions according to our two conditions. First, the lightweight hash function has a 256-bit output size. If the output size is less than 256-bit, its security strength is too small. The practice case of the attack to SHA-1, which has a 160-bit output size, was completed successfully by SHAttered [32]. Second, because the proposed architecture focuses on resource-constrained devices such as IoT devices, they are compactly designed to perform their specific purpose. Therefore, they do not have sufficient memory or power to calculate area or implement a hardware module because the device can perform a specific purpose. The area for implementing the lightweight hash function is under the 5000 GEs, which is less than the area of the general hash function. In real-case, some microchips generally used in IoT, such as MSP and ARM processors, are providing cryptographic hash engine (e.g., SHA-1 and SHA-2) that implements in hardware and the area for implementation of SHA-1 and SHA-2 are approximately 5500 GEs and 10,900 GEs, respectively [33]. Cryptolux [34] provides a list of lightweight hash functions proposed recently.

We select three hash functions in their hash list: QUARK, PHOTON, and SPONGENT; it can also be found in Table 5. These hash functions have different features in the perspective of security and performance. For example, even though QUARK shows the best throughput in the hash list, it has lower resistance than other hash functions. However, it does not mean this hash function is not secure. These hash functions have sufficient security level against cryptographic attacks. The hash list can support that block mining process quickly.

Table 5. Comparison security and performance of lightweight hash algorithms [34] (output size: 256 bit; GE < 5000; frequency: 100 kHz).

Hash	Ref.	Security			Performance		
		Pre	2nd Pre	Col	Area (GE)	Throughput (kbps)	Power (μW)
QUARK	CHES 10	224	112	112	4640	50	8.39
PHOTON	CRYPTO 11	224	128	128	4362	20.51	-
SPONGENT	CHES 11	240	128	128	3281	11.43	6.62

3.2.2. Flexible Hashchain: Hashchain Using Various Hash Function

The block structure in our architecture is composed of a header and body similar to the generally used block structure in the blockchain. The block header contains various fields, such as a block version number, a timestamp, a block size, and several transactions. Also, the header contains the hash of the previous block calculated from a hash function of the previous block header. The block body contains all the target transaction list. Each transaction generates a hash value, and then two adjacent hash values continue to implement a hash algorithm to generate a unique Merkle root. The block body and header are connected with the Merkle root. The nonce field is used for the proof of work algorithm. Miners try to find a nonce that generates a hash with a value lower than or equal to that set by network difficulty. A difficulty target is a number that regulates block creation time. However, in the proposed blockchain architecture, each block connected by the hash value is generated by a different hash function, which is selected according to the number of transactions. Due to this feature, in contrast to the general block structure, we add a field ("hashhelpher") for representing selected hash function from the hash list; this field is referred to in the mining and validation processes for checking which hash function is used. You can see our block structure in Figure 2.

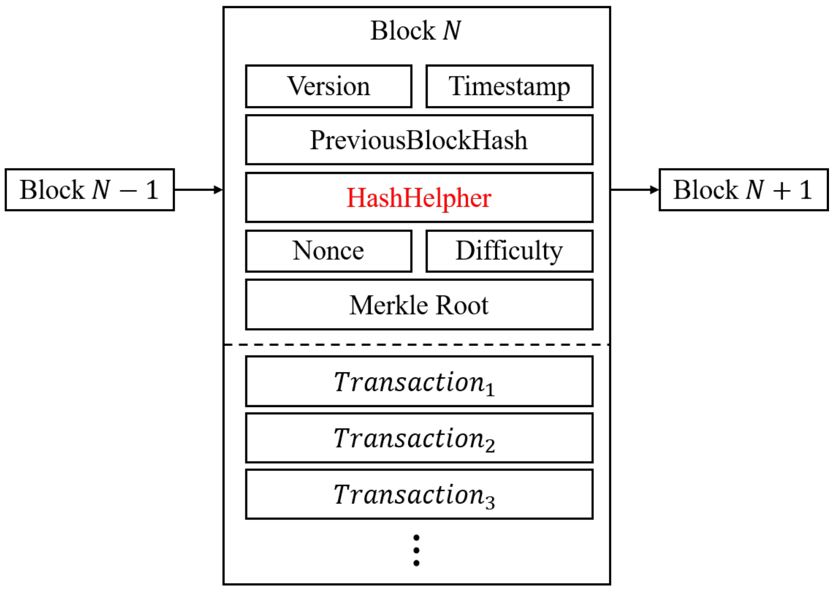

Figure 2. Block structure.

3.3. Network Interaction Process

In the proposed architecture, the blockchain network is located between the field Layer and the control Layer of the IIoT network; the blockchain network consists of the cell node and storage node, as mentioned before. These nodes transfer the data between field devices and control devices like a bridge. In the transferring process, they perform block mining and updating for data integrity. In the network, the interaction process is separated into two parts (Device Registration Process and Data Transferring Process), please find the entire interaction process in Figure 3. The detailed description of each process is as follows.

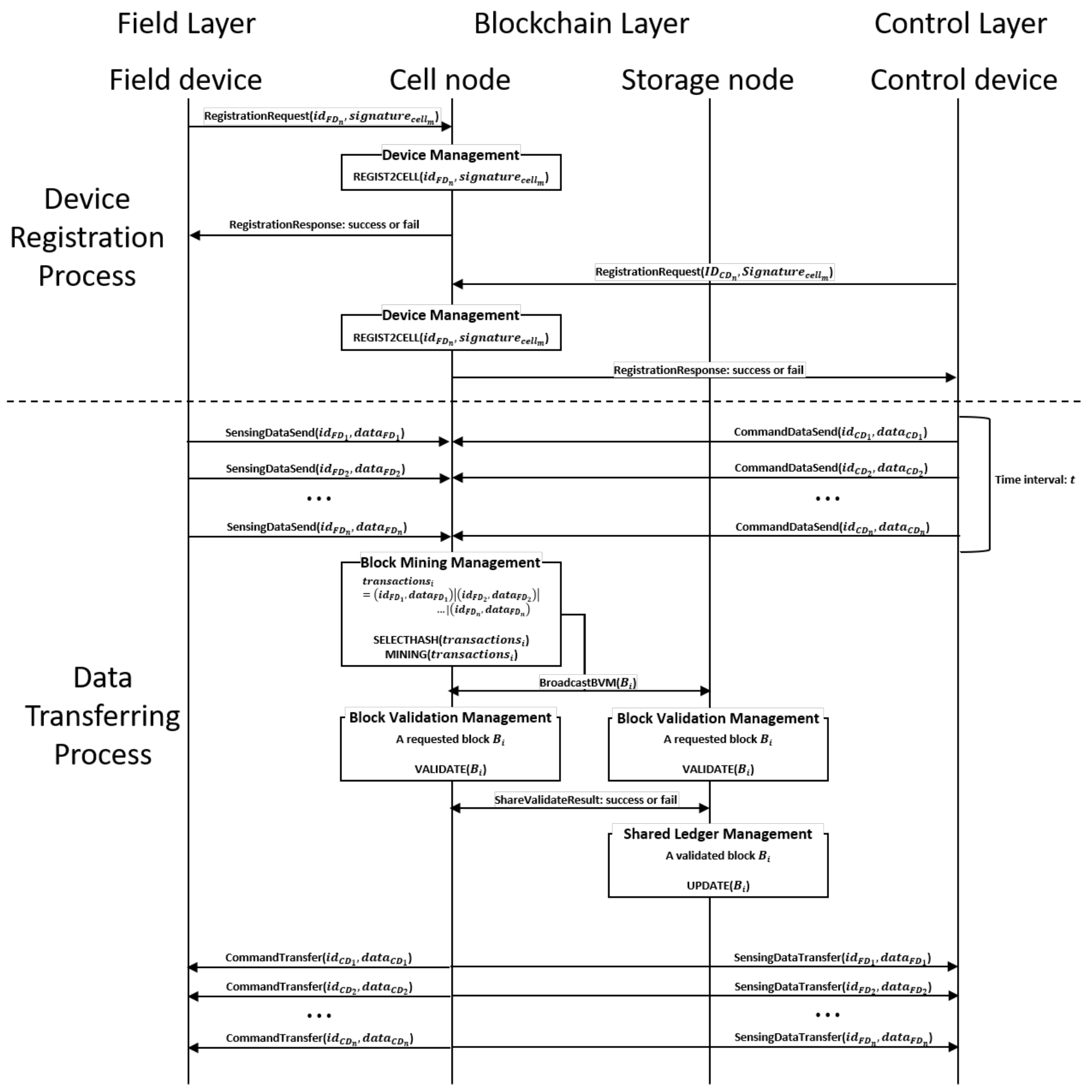
Field Layer
Blockchain Layer
Control Layer
Field device
Cell node
Storage node
Control device
Device Registration Process
RegistrationRequest(id_{FD_n}, $signature_{cell_m}$)
Device Management
REGIST2CELL(id_{FD_n}, $signature_{cell_m}$)
RegistrationResponse: success or fail
RegistrationRequest(ID_{CD_n}, $Signature_{cell_m}$)
Device Management
REGIST2CELL(id_{FD_n}, $signature_{cell_m}$)
RegistrationResponse: success or fail
Data Transferring Process
SensingDataSend(id_{FD_1}, $data_{FD_1}$)
SensingDataSend(id_{FD_2}, $data_{FD_2}$)
SensingDataSend(id_{FD_n}, $data_{FD_n}$)
CommandDataSend(id_{CD_1}, $data_{CD_1}$)
CommandDataSend(id_{CD_2}, $data_{CD_2}$)
CommandDataSend(id_{CD_n}, $data_{CD_n}$)
Time interval: t
Block Mining Management
$transactions_i = (id_{FD_1}, data_{FD_1})|(id_{FD_2}, data_{FD_2})| \dots |(id_{FD_n}, data_{FD_n})$
SELECTHASH($transactions_i$)
MINING($transactions_i$)
BroadcastBVM(B_i)
Block Validation Management
A requested block B_i
VALIDATE(B_i)
Block Validation Management
A requested block B_i
VALIDATE(B_i)
ShareValidateResult: success or fail
Shared Ledger Management
A validated block B_i
UPDATE(B_i)
CommandTransfer(id_{CD_1}, $data_{CD_1}$)
CommandTransfer(id_{CD_2}, $data_{CD_2}$)
CommandTransfer(id_{CD_n}, $data_{CD_n}$)
SensingDataTransfer(id_{FD_1}, $data_{FD_1}$)
SensingDataTransfer(id_{FD_2}, $data_{FD_2}$)
SensingDataTransfer(id_{FD_n}, $data_{FD_n}$)

Figure 3. Proposed architecture interaction process.

The device registration process is a process that the requested device link to the blockchain network. First, the device sends the registration request with their identification number (id_{FD_n}: identification number of n-th field device; id_{CD_n}: identification number of n-th control device) and pre-shared signature of cell which is nearly located ($signature_{cell_n}$: n-th cell signature). The identification number and cell signature are unique identifiers that are assigned to each device and the cell. It is used to figure out what devices are included in a cell, and this information is stored in device manager of cell node as a list ($list_DEV$). Then, the cell node, which received a registration request from devices performs the device management function of the device manager. In this process, the device manager compares the pre-shared signature with its cell signature ($signature_{own}$). If the signature is matched, the device manager adds the device identification number to the device list. Algorithm 1 represents the device registration process

Algorithm 1: Device Management

```
Input   : $id_{FD_n}, signature_{cell_n}$
Output  : success or fail
1  Procedure REGIST2CELL($id_{FD_n}, signature_{cell_n}$)
2      if $signature_{cell_n} = signature_{own}$ then
3          $list_{DEV} \leftarrow list_{DEV} | id_{FD_n}$
4          return success
5      else
6          return fail
7      end if
8  End Procedure
```

The data transferring process is a process where the cell node gathers the data needing to be processed from devices and performs block mining, block validation, and updating of the ledger in storage node. First, in this process, the cell node gathers the sensing data from field devices, also gathers the control command from control devices in a predefined time interval (t). Then, the cell node process the gathered data to transactions ($transactions_i$) and performs the block mining management function. Block mining management consists of 2 procedures. First one is the selection of lightweight hash function. The hash function is selected by comparing the number of transaction to the predefined threshold (h_1, h_2). Another one is the mining process that finds a nonce that satisfies target difficulty by calculating the hash value ($Hash(B_i)$) to the block (B_i). This procedure is basically the same "PoW". When the right nonce is founded, the cell node requests validation of the block to other nodes in the blockchain network ($BroadcastBVM(B_i)$), and waits for the response which contains the result of validation. Besides, other nodes that received a block validation request perform the block validation management function, and then check whether a requested block is correctly synchronized and whether the nonce satisfies the target difficulty. The synchronization checking is performed by comparing the previous hash value of the requested block to the stored previous hash value in the node, and the nonce checking is performed by calculating the hash value. If the validation result is true, the block validation management broadcasts the result, and the stored previous hash value is updated. You can see the block mining management process in Algorithm 2. Also, please see the validation management process in Algorithm 3.

Next, after the block validation, the block update management function is performed in the storage node. The storage node checks its capacity, whether it has a sufficient area to append the validate block. If the storage node has a sufficient area, the validated block is appended to the ledger (L). However, if the storage node lacks the area to store the block, the old part of the ledger is deleted ("DeleteOld(L)") and a validated block is appended. You can see the block update management process in Algorithm 4. When all of these processes are completed, each sensing data and control command are delivered to their destinations.

Algorithm 2: Block Mining Management

```
Input  : $transactions_i = (id_{FD_1}, data_{FD_1})|(id_{FD_2}, data_{FD_2})|...|(id_{FD_n}, data_{FD_n})$
Output : success or fail
Procedure SELECTHASH($transactions_i$)
    if $sizeof(transactions_i) < h_1$ then
        Hash ← SPONGENT
    else if $h_1 < sizeof(transactions_i) < h_2$ then
        Hash ← PHOTON
    else
        Hash ← QUARK
    end if
End Procedure
Procedure MINING($transactions_i$)
    Initialize($B_i$)
    $B_i$.PreviousBlockHash ← PreviousBlockHash
    $B_i$.HashHelper ← Hash
    $B_i$.Nonce ← 0
    while Hash($B_i$) < difficulty do
        $B_i$.Nonce++
    endw
    $v$ ← BroadcastBVM($B_i$)
    if $v = true$ then
        PreviousBlockHash ← Hash($B_i$)
        return success
    else
        return fail
    end if
End Procedure
```

Algorithm 3: Block Validation Manager

```
Input  : a requested block $B_i$
Output : true or false
Procedure VALIDATE($B_i$)
    if $B_i$.PreviousBlockHash $\neq$ PreviousBlockHash then
        return false
    Hash ← $B_i$.HashHelper
    difficulty ← $B_i$.difficulty
    if Hash($B_i$) < difficulty then
        PreviousBlockHash ← $B_i$.PreviousBlockHash
        if storage node then
            Update($B_i$)
        return true
    else
        return false
    end if
End Procedure
```

Algorithm 4: Block Update Management

```
    Input   : a validated $B_i$
    Output  : a updated shared ledger $L$
 1  Procedure Update($B_i$)
 2      if sizeof($L + B_i$) < capacity then
 3          $L \leftarrow L \mid B_i$
 4          return $L$
 5      else
 6          $L \leftarrow$ DeleteOld($L$)
 7          $L \leftarrow L \mid B_i$
 8          return $L$
 9      end if
10  End Procedure
```

4. Analysis

4.1. Simulation Results

In the proposed blockchain architecture, each block is connected by different lightweight hash functions, which are selected from the lightweight hash list according to the number of transactions. As previously mentioned, the lightweight hash list consists of three algorithms (QUARK, PHOTON, and SPONGENT) which have different features in terms of cryptographic security and performance. Following this reason, the whole processing time of the network is changed by the selected hash function. For the simulating our proposed architecture, we build an equation that can approximately calculate the whole processing time. We supposed that the parameters which can exist in our proposed architecture are as follows (the processing time is T (sec), the network communication delay is d_n, the difficulty of the block mining is D, the throughput (kbps) of the hash function is s, the size of transactions (bytes) is W the validate communication delay is d_v, the updating delay for storing the block is d_u, and the thresholds for changing hash function are h_1 and h_2, respectively). Then, we can get the approximated processing time using Equation (1).

$$\begin{aligned} T \;&=\; d_n + D\frac{W \times 8}{s \times 1000} + d_v\frac{W \times 8}{s \times 1000} + d_u W, \text{ where } D < h_1 : s = 11.43 \text{ (SPONGENT)}, \\ &h_1 < D < h_2 : s = 20.51 \text{ (PHOTON)}, D > h_2 : s = 50 \text{ (QUARK)}. \end{aligned} \tag{1}$$

Following the above equation, we first calculated the processing time of each three hash functions without changing the hash function. When we calculate the processing time, we supposed that the difficulty changed from 1 to 10, and the size of data is changed from 7 to 260. The difficulty is a minimum exponentiation value of the trials, 2^x, for finding the right hash value; we thought it did not need to be too reputable, such as bitcoin, because the IIoT may be constructed by private blockchain, and not the public blockchain. We supposed that the size of data is packet size of Modbus-TCP that has a range of 7 to 260 bytes. Also, we supposed that the delay does not exist in the network ($d_n = 0, d_v = 0, d_u = 0$). Figure 4 represents the results. As you can see, if the difficulty increases, the processing time increases exponentially, and if the size of data increases, the processing time increases linearly. To figure out the effect of changing the hash function, we simulated the processing time with the fixed difficulty ($D = 9$), and we supposed that the multiple packets are processed. You can see the result in Figure 5, also note that the processing time decreases when it reaches a certain level. However, for performing similar to the simulated result, the threshold h_1, h_2 is carefully selected.

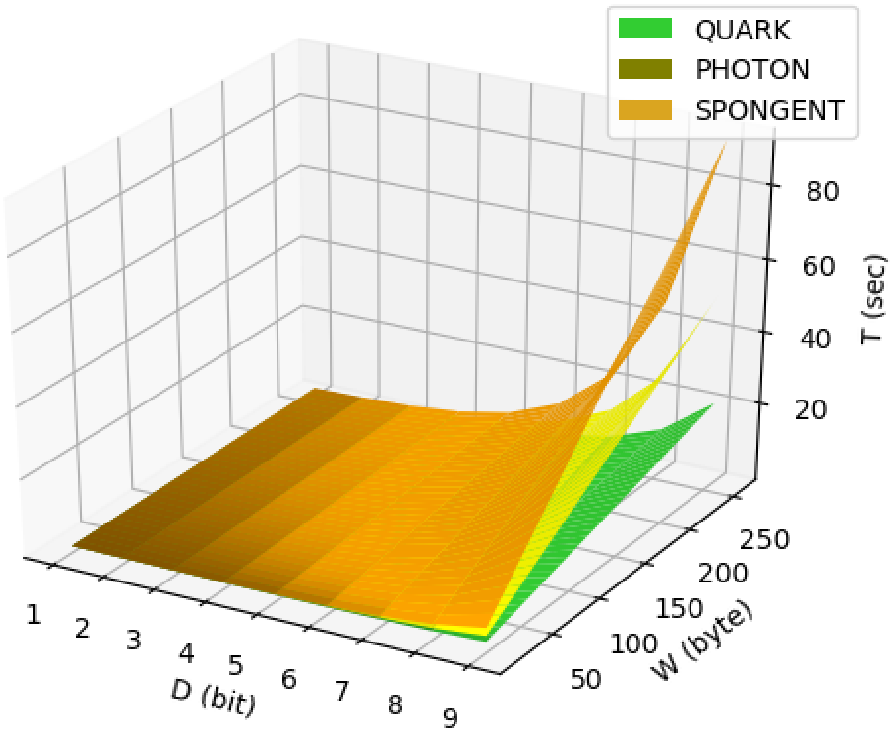

Figure 4. Processing time of QUARK, PHOTON, and SPONGENT.

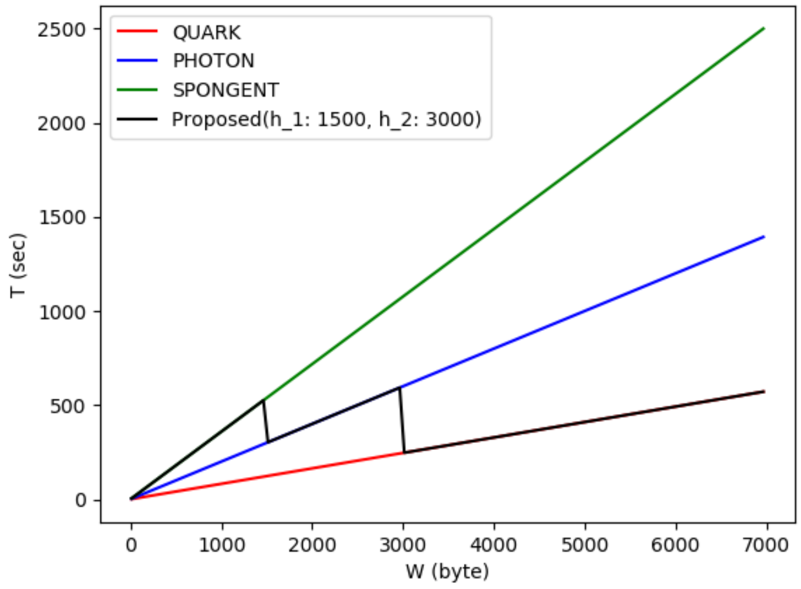

Figure 5. Result of changing hash function.

Additionally, we simulated a situation where the hash function is implemented redundantly for parallel computation. In reality, IIoT technology has various applications, and each application has different requirements: there is much difference in terms of latency and update frequency. For example, in monitoring and supervision field, the gas detection or pressure sensor requires the latency under 1 s, and the vibration sensor requires the latency from a few seconds to a few days. In the closed-loop control field, the required latency of sensors is less than 500 milliseconds. In the interlocking and control field, the latency is more restricted, and it is less than 250 milliseconds [35]. In the simulation, we assume that the multiple QUARK module is implemented by hardware and compared the processing time, see the result in Figure 6. In our simulation, if the hash function is multiple implemented, the processing time decreases linearly to the number of the module parallel implemented (p). The simulated processing time is about 2 seconds when the five QUARK modules are implemented. Based on results, this architecture can be applied to various IIoT applications, especially vibration sensor and temperature sensor in the monitoring and supervision field. Also, if the lightweight hash functions that can operate with better performance are developed, we argue that this architecture can be applied to other industrial IoT applications, which requires latency to be under 1 s.

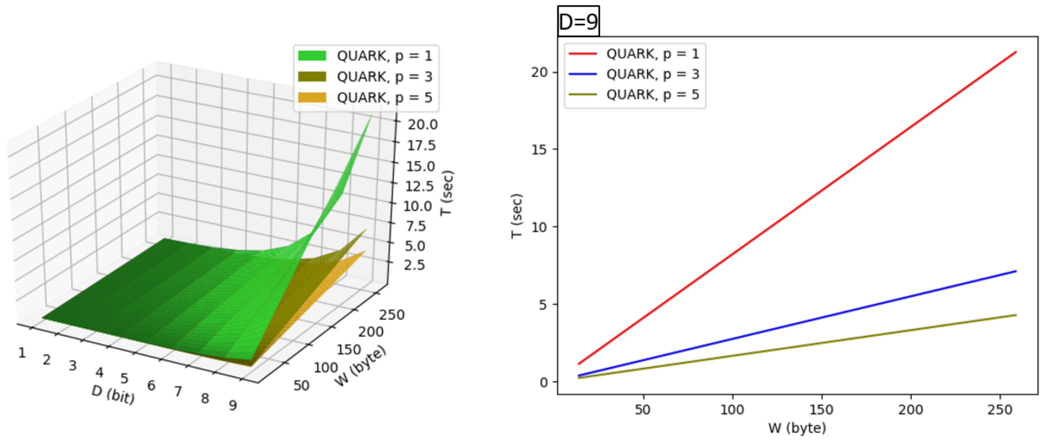

Figure 6. Result of parallel QUARK: (Left) - non-fixed parameter, (Right) - fixed parameter $D = 9$.

However, if we implement the hash function parallel, it causes additional costs. The costs for the parallel implementation of hash functions consist of two parts. The first is an implementation cost, and the second is an operation cost. The hash functions require some area (GEs) in hardware implementation, and the implementation cost means the required area for parallel implementing. In our case, each selected hash functions requires less than 5000 GEs (see Table 5), and the total area for the three hash functions is 12,283 GEs. However, if the hash functions are implemented parallel, the required area for implementing will be increased linearly, depending on p. For example, if the three hash functions implemented doubly, the total area for implementing is 24,566 GEs. The operation cost means that the power consumption for calculating hash values in the mining process. When the hash functions are implemented parallel, it makes the hash modules calculate simultaneously the hash value, which satisfies the target difficulty in the mining process. It does not mean that the total amount of calculation reduced. Each hash modules require equal power separately. Therefore, the power consumption for searching the right hash value increases linearly; thereby, the operation cost will increase linearly similar to the implementation cost, depending on p. Because of this trade-off between processing time and cost, p has to be selected according to their resources. We do not recommend parallel implementation for devices with a highly restricted battery or area.

4.2. Availability of Proposed Architecture

In Section 2.3, we introduced three considerations including Computational resources, Latency, and Scalability. The explanation of how can we satisfy each consideration is as follows.

1. Lightweight hash for computational resources: IIoT network consists of IoT devices with limited resources, and the resource-constrained devices have a tiny area and low power. For reducing the computational burden of a consensus protocol, we selected the lightweight hash functions QUARK, PHOTON, and SPONGENT. These hash functions were implemented with a small area (GEs < 5000), and also consume little power (μW < 10). It can be easily applied to resource-constrained devices, in contrast with existing hash functions such as SHA-256. Therefore, the consensus protocol using these lightweight hash function can be performed more lightly.
2. Hash list for latency: The mining process in the typical blockchain is performed with a predefined hash function for searching the specific hash value. However, in the proposed architecture, the cell

node is responsible for the mining process to select a hash function from the hash list according to the network traffic. The hash functions in the hash list have different performances in terms of throughput and security. In the case of throughput, they show good throughput, and are listed in descending order: QUARK, PHOTON, then SPONGENT. Oppositely, in the case of security, including pre-image resistance, second pre-image resistance, and collision resistance, they show good security in order SPONGENT, PHOTON, QUARK. If the transactions increase according to network traffic, the cell node changes the hash function to one that has better throughput. Even though this also means that the security level of hash function decreases, it does not mean these hash functions are not secure. This approach can reduce the latency of the mining and updating process.

3. Composing cell node for scalability: In the industrial field, field devices can be spread in a wide location. The broadcasting for block validation can cause scalability issue because IoT devices have low bandwidth. To reduce the burden of broadcasting, we composed the cell nodes, which are nearby located to field devices and control devices. In the proposed architecture, each cell node gathers the transactions from the adjacent field devices, control devices. Also, it processes the mining process, and then broadcasts the block to other cell nodes participating in the blockchain.

5. Conclusions

In this paper, we proposed blockchain architecture that can change the hash function of blockchain flexibly according to the amount of transactions improving the availability of the blockchain network. In this architecture, we selected three lightweight hash functions (QUARK, PHOTON, and SPONGENT), which show better performance in perspective of implementing area, throughput, and power consumption. These hash functions can ensure cryptographic security and be implemented using small area (under 5000 GEs) for resource-constrained devices. Then, using these hash functions, we connected each data block by flexible hashchain. This approach can reduce the computational burden and latency. Also, we separated the field to some cells and composed the cell nodes to control each cell. This approach can improve the scalability of the network. In our simulation results, we revealed that the proposed architecture is suitable for an environment that should be processed within certain times, especially monitoring and supervision field among IIoT applications. Additionally, if the hash functions are implemented parallel or more lightweight hash functions are developed, we argue that the proposed architecture can be applied more various IIoT applications, which require the latency under 1 s. However, there are many IIoT applications that require the restricted latency (less than 500 ms), such as closed-loop control field, interlocking, and control field. In these cases, it seems that the availability still remains the main concern when blockchain technology is applied.

In our future work, to demonstrate the performance of our proposed architecture on various IIoT applications, we plan to conduct a simulation using NS-3 simulator that can simulate similarly to the real environment. Furthermore, we will implement our proposed architecture in actual devices for analyzing performance and security against existing blockchain attacks and ICS attacks.

Author Contributions: Conceptualization, B.S.; methodology, B.S. and J.P.; investigation, J.P.; writing—original draft preparation, B.S.; project administration, J.H.P.; funding acquisition, J.H.P.

Acknowledgments: This study was supported by the Advanced Research Project funded by the SeoulTech (Seoul National University of Science and Technology).

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

References

1. Nakamoto, S. Bitcoin: A Peer-To-Peer Electronic Cash System. 2008. Available onlie: http://bitcoin.org/bitcoin.pdf (accessed on 6 September 2019).
2. Sharma, P.K.; Moon, S.Y.; Park, J.H. Block-VN: A Distributed Blockchain Based Vehicular Network Architecture in Smart City. *JIPS* **2017**, *13*, 184–195.

3. Nguyen, G.T.; Kim, K. A Survey about Consensus Algorithms Used in Blockchain. *J. Inf. Process. Syst.* **2018**, *14*, 101–128.
4. Kim, H.W.; Jeong, Y.S. Secure authentication-management human-centric scheme for trusting personal resource information on mobile cloud computing with blockchain. *Hum.-Cent. Comput. Inf. Sci.* **2018**, *8*, 11. [CrossRef]
5. Khan, M.A.; Salah, K. IoT security: Review, blockchain solutions, and open challenges. *Future Gener. Comput. Syst.* **2018**, *82*, 395–411. [CrossRef]
6. Gilchrist, A. *Industry 4.0: The Industrial Internet of Things*; Apress: New York, NY, USA, 2016; pp. 8–9.
7. Fortino, G.; Savaglio, C.; Zhou, M. Toward opportunistic services for the industrial Internet of Things. In Proceedings of the 2017 13th IEEE Conference on Automation Science and Engineering (CASE), Xi'an, China, 20–23 August 2017; pp. 825–830.
8. Bahga, A.; Madisetti, V.K. Blockchain platform for industrial internet of things. *J. Softw. Eng. Appl.* **2016**, *9*, 533. [CrossRef]
9. Sharma, P.K.; Ryu, J.H.; Park, K.Y.; Park, J.H.; Park, J.H. Li-Fi based on security cloud framework for future IT environment. *Hum. Cent. Comput. Inf. Sci.* **2018**, *8*, 23. [CrossRef]
10. Wan, J.; Li, J.; Imran, M.; Li, D. A Blockchain-Based Solution for Enhancing Security and Privacy in Smart Factory. *IEEE Trans. Ind. Inf.* **2019**, *15*, 3652–3660. [CrossRef]
11. Xu, H.; Yu, W.; Griffith, D.; Golmie, N. A Survey on Industrial Internet of Things: A Cyber-Physical Systems Perspective. *IEEE Access* **2018**, *6*, 78238–78259. [CrossRef]
12. Fernández-Caramés, T.M.; Fraga-Lamas, P. A Review on the Use of Blockchain for the Internet of Things. *IEEE Access* **2018**, *6*, 32979–33001. [CrossRef]
13. Novo, O. Blockchain meets IoT: An architecture for scalable access management in IoT. *IEEE Int. Things J.* **2018**, *5*, 1184–1195. [CrossRef]
14. Dorri, A.; Kanhere, S.S.; Jurdak, R.; Gauravaram, P. Blockchain for IoT security and privacy: The case study of a smart home. In Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Hawaii, HI, USA, 13–17 March 2017; pp. 618–623.
15. Dorri, A.; Kanhere, S.S.; Jurdak, R. Towards an optimized blockchain for IoT. In Proceedings of the Second International Conference on Internet-of-Things Design and Implementation, Pittsburgh, PA, USA, 18–21 April 2017; pp. 173–178.
16. Biswas, K.; Muthukkumarasamy, V. Securing smart cities using blockchain technology. In Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, Australia, 12–14 December 2016; pp. 1392–1393.
17. Sukhwani, H.; Martínez, J.M.; Chang, X.; Trivedi, K.S.; Rindos, A. Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric). In Proceedings of the 2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS), Hong Kong, 26–29 September 2017; pp. 253–255.
18. Boudguiga, A.; Bouzerna, N.; Granboulan, L.; Olivereau, A.; Quesnel, F.; Roger, A.; Sirdey, R. Towards better availability and accountability for iot updates by means of a blockchain. In Proceedings of the 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), Paris, France, 26–28 April 2017; pp. 50–58.
19. Li, R.; Song, T.; Mei, B.; Li, H.; Cheng, X.; Sun, L. Blockchain for large-scale internet of things data storage and protection. *IEEE Trans. Serv. Comput.* **2018**. [CrossRef]
20. Samaniego, M.; Deters, R. Hosting virtual iot resources on edge-hosts with blockchain. In Proceedings of the 2016 IEEE International Conference on Computer and Information Technology (CIT), Nadi, Fiji, 7–10 December 2016; pp. 116–119.
21. Samaniego, M.; Deters, R. Using blockchain to push software-defined IoT components onto edge hosts. In Proceedings of the International Conference on Big Data and Advanced Wireless Technologies, Blagoevgrad, Bulgaria, 10–11 November 2016; ACM: New York, NY, USA, 2016; p. 58.
22. Sharma, P.K.; Singh, S.; Jeong, Y.S.; Park, J.H. Distblocknet: A distributed blockchains-based secure sdn architecture for iot networks. *IEEE Commun. Mag.* **2017**, *55*, 78–85. [CrossRef]
23. Liu, Y.; Wang, K.; Lin, Y.; Xu, W. LightChain: A Lightweight Blockchain System for Industrial Internet of Things. *IEEE Trans. Ind. Inf.* **2019**, *15*, 3571–3581. [CrossRef]

24. Percival, C. Stronger Key Derivation via Sequential Memory-Hard Gunctions. 2009. Available online: https://www.tarsnap.com/scrypt/scrypt.pdf (accessed on 6 September 2019).
25. Litecoin Project Community, About LiteCoin, 2018. Available online: https://litecoin.org/ (accessed on 6 September 2019)
26. Ray, J. Ethereum (Whitepaper), 2018. Available online: https://github.com/ethereum/wiki/wiki/White-Paper (accessed on 6 September 2019).
27. Bertoni, G.; Daemen, J.; Hoffert, S.; Peeters, M.; Van Assche, G.; Van Keer, R. Kessak, 2008. Available online: https://keccak.team/ (accessed on 6 September 2019).
28. Cryptorival.com. Cryptocurrency Algorithms, 2019. Available online: https://cryptorival.com/algorithms (accessed on 6 September 2019).
29. Choi, S.; Sun, K.; Eom, H. Resource-Efficient Multi-Source Authentication Utilizing Split-Join One-Way Key Chain. In *Emerging Trends in ICT Security*; Elsevier: Amsterdam, The Netherlands, 2014; pp. 267–279.
30. Reyna, A.; Martín, C.; Chen, J.; Soler, E.; Díaz, M. On blockchain and its integration with IoT. Challenges and opportunities. *Future Gener. Comput. Syst.* **2018**, *88*, 173–190. [CrossRef]
31. Williams, T.J. The Purdue enterprise reference architecture. *Comput. Ind.* **1994**, *24*, 141–158. [CrossRef]
32. shattered.io. SHAttered, 2019. Available online: https://shattered.io/ (accessed on 6 September 2019).
33. Guo, J.; Peyrin, T.; Poschmann, A. The PHOTON family of lightweight hash functions. In Proceedings of the 31st Annual Conference on Advances in Cryptologym, Santa Barbara, CA, USA, 14–18 August 2011; pp. 222–239.
34. Cryptolux.org. Summary of the Main Characteristics of the Current Lightweight Hash Functions. 2019. Available online: https://www.cryptolux.org/index.php/Lightweight_Hash_Functions#cite_note-BKLT11-22 (accessed on 6 September 2019).
35. Nikoukar, A.; Raza, S.; Poole, A.; Güneş, M.; Dezfouli, B. Low-power wireless for the internet of things: Standards and applications. *IEEE Access* **2018**, *6*, 67893–67926. [CrossRef]