**SPECIAL SECTION ON GREEN COMMUNICATIONS ON WIRELESS NETWORKS**

Received February 4, 2020, accepted February 18, 2020, date of publication February 24, 2020, date of current version March 3, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2975920*

Blockchain-Based Distributed Firmware Update Architecture for IoT Devices

SEOYUN CHOI AND JONG-HYOUK LEE, (Senior Member, IEEE)



Protocol Engineering Laboratory, Department of Software, Sangmyung University, Cheonan 31066, South Korea

Corresponding author: Jong-Hyouk Lee (jonghyouk@smu.ac.kr)

This work was supported by the research grant from Sangmyung University.

 **ABSTRACT** The Internet of Things (IoT) which creates a hyper-connected society is playing a majorrole in the 4th industrial revolution. The IoT is being leveraged across various elds of business globally and the number of IoT devices is causing serious security concerns. Since the rmware update of an IoT device is necessary for its lifecycle, secure rmware update of the IoT device is being brought as the rst step in IoT security. The Internet Engineering Task Force (IETF) Software Updates for Internet of Things (SUIT) working group has started to specify a software update architecture for IoT devices. However, the current SUIT working group adopts a traditional client-server model to distribute rmware images, which potentially causes security risks. The current approach of the SUIT working group is unable to solve a targeting issue and an author-disappearing issue, which is suggested in this paper. Therefore, in this work, we introduce a distributed rmware update architecture based on the SUIT rmware update architecture applying blockchain. Our update architecture can prevent the issues we concern through the characteristics of blockchain, such as decentralization, transparency, and irreversibility. The blockchain network has registration nodes that process registration of manifest and rmware image les from authors, and retrieval nodes that process downloading manifest and rmware image les. The rmware image les are stored in a distributed le system and the hash values of rmware image chunks are stored on the blockchain with manifest les. The proposed architecture in this paper enables the irreversible downloads even in the author-disappearing state and tolerant to a single point of failure.



 **INDEX TERMS** Author-disappearing issue, blockchain, rmware update, Internet of Things (IoT).



**I. INTRODUCTION**

The Internet of Things (IoT) implies the status of network connection for interactions between things of human beings via embedded communication systems. The IoT is used in a very wide range of elds as it provides close connec-tions between things and humans. Even though IoT tech-nology is being expeditiously improved as a key technology in Industry 4.0, the security risks of IoT devices are con-cerned. IoT devices operate with low computing power using micro controller units (MCUs) for weight lightening, but this lightweight computing characteristic of the IoT device accompanies security limits.

A rmware update offers a new service or security patch to IoT devices. Since it is necessarily required for the IoT device, the recent cyber-attacks are focused on the rmware level. As the result, numerous researches for secure rmware

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Pau .



updates have been conducted recently [1], [2]. The rmware update failures occur due to network issues or cyber-attacks. The supports for integrity and authentication of the rmware images thus required. The Software Updates for Internet of Things (SUIT) working group of the Internet Engineering Task Force (IETF) is on a process of creating standards for an IoT rmware update architecture and rmware manifest le. The traditional client-server model that uses a centralized database which is an easy and high-value target for various kinds of attacks. An attack on a server can violate the avail-ability of all data stored on the server. Also, rmware updates for IoT devices are vulnerable against an author-disappearing issue that the IoT device manufacturers or rmware ven-dors are unable to provide rmware updates in time due to cyber-attacks or disappearing due to their funding problems.

Blockchain ensures that every node stores the same data based on an append-only distributed ledger, which provides integrity, decentralization, and irreversibility. In this paper, we propose a distributed SUIT rmware update architecture,

|  |  |  |  |
| --- | --- | --- | --- |
| 37518 | This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/ | VOLUME 8, 2020 |  |
|  |  |

S. Choi, J.-H. Lee: Blockchain-Based Distributed Firmware Update Architecture for IoT Devices

which is based on blockchain. The proposed architecture is designed to provide a secure rmware update and also to address the author-disappearing issue.

The remainder of this paper is as follows: In Section [II,](#page2) the current SUIT architectures, blockchain, and exist-ing blockchain-based rmware update architectures are explained. In Section [III,](#page4) we introduce our blockchain-based distributed rmware update architecture. Then, we present comparison results of the proposed architecture with the existing blockchain-based rmware update architecture in Section [IV.](#page7) In Section [V,](#page8) we conclude this paper.

**II. RELATED WORK**

Existing rmware updates are mostly executed in client-server architecture form. A server takes a charge of centralized management of the whole data in a client-server model. This leads a server to be a vulnerable target for an adversary. In addition, if excessive request from rapidly increasing IoT devices overwhelm the server, update processes can be failed. In recent years, blockchain, a distributed ledger offering decentralization, integrity and irreversibility of data has been considered as a mean to solve the limitations of client-server architecture and guarantee integrity of the rmware images.

**A. BLOCKCHAIN**

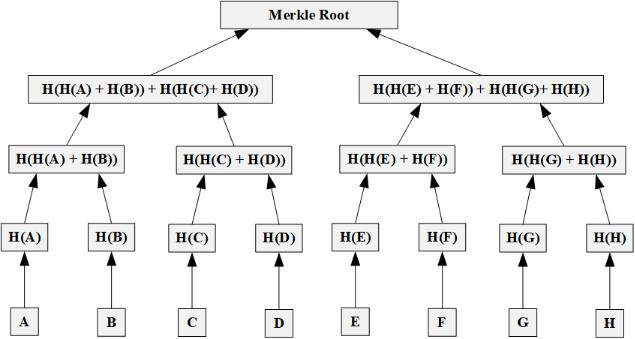
The concept of blockchain rst appeared in the Bitcoin white paper written by Satoshi Nakamoto in 2008 [3]. Blockchain technology has gained substantial attention with a departure from the previous centralized network model. It enables that every participating node can verify, append, and store the data without any centralized entities. After the appearance of the Bitcoin, many kinds of cryptocurrencies have emerged based on blockchain [4]. The researches on security threats and mit-igations of the blockchain are being actively conducted [5].

The operation process of blockchain in the Bitcoin system is as follows. First, when a new transaction has occurred, it is broadcasted to every node. Then, full nodes verify the transaction with a digital signature. After the veri cation, the full nodes store the transaction in their memory pools. Then, the full nodes decide one block at regular intervals through a result of an agreed consensus algorithm [6]. The result of the consensus algorithm is then broadcasted to all nodes in the Bitcoin system, and the nodes link the result (i.e., the new block) to their own ledger. In the respect that every node contributes to consensus for creating new blocks and maintaining the same distributed ledger, blockchain ensures the transparency of transactions.

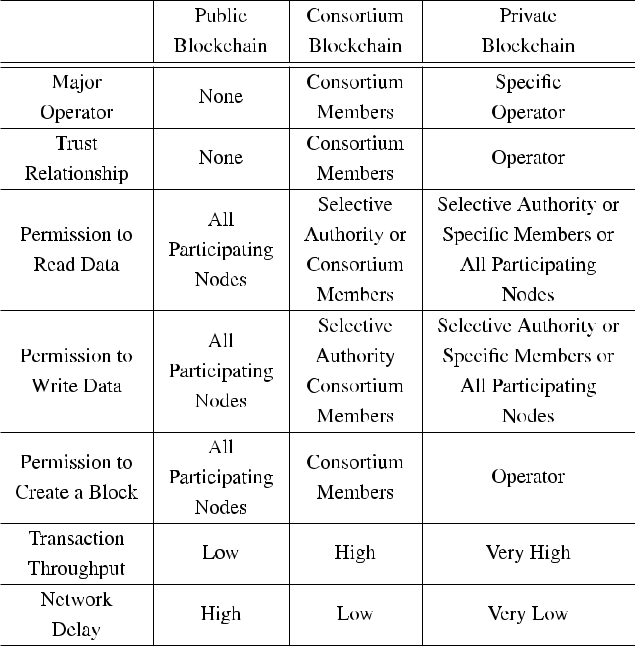
Every block has an own hash value for its identi cation and each block is linked through a previous block hash stored in the block header. Transaction IDs are paired and hashed, constructing a merkle tree. The root hash maintained in a block header and guarantees integrity of whole transactions stored in the block. Fig. [1](#page2) shows an example of a merkle tree structure ensuring integrity of the transactions in the Bitcoin system [7].

VOLUME 8, 2020

**FIGURE 1.** Merkle tree.



**TABLE 1.** Architecture types of blockchain networks.



**B. ARCHITECTURE TYPES OF BLOCKCHAIN**

The blockchain platform can be divided into permissoinless and permissioned [8]. The permissionless platform includes a public blockchain (e.g., Bitcoin) in which every consisting node can participate to the network. The permissioned plat-form includes a private blockchain (e.g., Hyperledger Fabric) and a consortium blockchain. The private blockchain has a structure that a speci c operator manages an authority service for all nodes. The consortium blockchain has a structure in which a plurality of organizations forms several private blockchain networks. Table [1](#page2) describes the features and illus-trates the public, consortium, and private blockchain network.

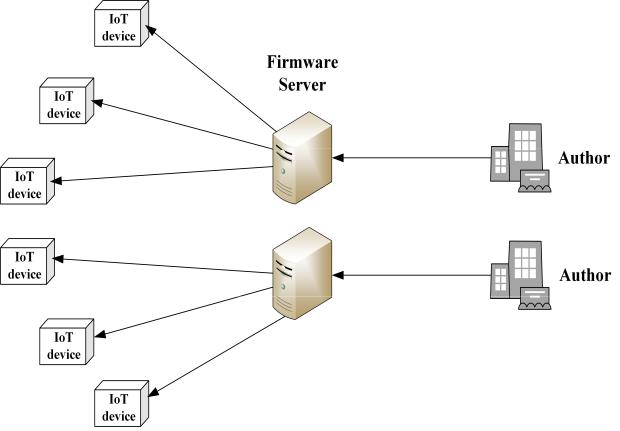
**C. SUIT FIRMWARE UPDATE ARCHITECTURE**

The IETF SUIT working group is developing the rmware update architecture for IoT devices and manifest les for rmware updates at the IETF standardization organiza-tion [9], [10]. A server that rmware and manifest les are

37519

S. Choi, J.-H. Lee: Blockchain-Based Distributed Firmware Update Architecture for IoT Devices

**FIGURE 2.** SUIT architecture adopting the client-server model.



uploaded is called a rmware server in the SUIT architec-ture. The manifest is a le that contains metadata about the rmware image. A manufacturer produces IoT devices or a vendor develops the rmware and manifest les are called authors.

The rmware update process is as follows: First, an author creates manifest and rmware le, signs them, and upload to the rmware server. The IoT device then queries the rmware server for the manifest and rmware le. When the rmware server sends the manifest and rmware le in order, the les veri ed each and downloaded.

If a rmware update is aborted or an IoT device fails to download the latest version of rmware, the IoT devices are considered as a vulnerable device due to lack of security services or secure patch, which may lead the IoT devices to be exposed to an attack. Therefore, the rmware update is a crucial security process for the IoT devices. However, the traditional client-server architecture leads to serious over-load. Plus, if a server is attacked or goes down, all clients that need to get a rmware image from the server are exposed to threats. Figure [2](#page3) shows the SUIT architecture that adopts the client-server model.

**D. BLOCKCAHIN-BAESD FIRMWARE UPDATES**

Here, we introduce and analyze the existing studies of blockchain-based rmware updates.

1) SCHEME 1 BY B. Lee AND J.-H. Lee

The paper written by Lee and Lee [1] was the rst paper that suggests a blockchain-based rmware update for IoT devices. The architecture suggested in the paper consists of request node, response nodes, and verifying nodes. A veri cation node is directly managed by a manufacturer and always aware of the latest version of the rmware. A request node sends its rmware version to a verifying node, then the verifying node compares the version of the rmware whether the requested rmware version is the latest. If the version of the requested device’s rmware is not the latest, the request node requests again to a neighbor node (besides verifying nodes). Between

37520

the request node and the response node, a node that has an older version of rmware downloads the rmware from the counterpart node. The repetition of this comparison process nally leads all nodes to be updated.

However, the architecture proposed in the paper has limi-tations that the blockchain network consists of IoT devices as nodes and applies a consensus algorithm that requires heavy resource consumption such as Proof-of-Work (PoW). As IoT devices operate with limited resources, it is dif cult to perform such a consensus algorithm and digital signature ver-i cation. In addition, the architecture only supports updates from one manufacturer, so a countermeasure for multiple vendors is demanded.

2) SCHEME 2 BY A. Boudguiga et al.

The paper was proposed in 2017 by Boudguiga *et al.* [11]. The update procedure suggested in this paper is as follows. A manufacturer selects the IoT devices to update and uploads the software to the manufacturer’s web portal. The IoT devices receive an update noti cation from the manufacturer.

The suggested architecture offers two methods of the rmware update. The rst method is that manufacturers upload the rmware binaries within the transaction and issue the transaction to the blockchain network. In this case, the size of blocks will get larger, but the rmware binaries are pre-served eternally. Another method is con guring innocuous-ness nodes which are very reliable (e.g., nodes af liated to a national agency or security company). In the second case, manufacturers do not put rmware binaries within the trans-action, but innocuousness nodes get the rmware binaries directly from manufacturers to check integrity or bugs of the rmware le.

However, this architecture has drawbacks that the block size becomes larger as the manufacturers transmit the rmware binaries in transactions to the blockchain network.

3) SCHEME 3 BY A. Yohan AND N. Lo

The paper was proposed by A. Yohan and N. Lo in 2018 [12]. A vendor repository and a broker repository are stor-ing rmware binaries and associated rmware information. A broker repository distributes the traf c overhead of vendor repositories. Vendor nodes, full nodes, and light nodes are constituting of a blockchain network for the rmware update. When a vendor repository announces the latest rmware to the vendor node, it deploys a smart contract requesting full nodes to verify the latest rmware. When the veri cation is completed, gateway nodes nd the devices belong to the manufacturer through the gateways and sends URI to the gateway. The gateway downloads the rmware binary from the vendor repository and distribute the rmware binary to the devices.

However, this architecture arouses concerns about exces-sive traf c and costs because the manufacturers must deploy smart contracts whenever they distribute new rmware les or duplicate rmware les to the broker nodes. In addition, an update failure may occur if the gateway is compromised.

VOLUME 8, 2020

S. Choi, J.-H. Lee: Blockchain-Based Distributed Firmware Update Architecture for IoT Devices

1. **DISTRIBUTED FIRMWARE UPDATE ARCHITECTURE**

The blockchain-based rmware update architecture brought up in this paper is based on the rmware architecture model currently being standardized by the IETF SUIT.

The proposed architecture contains a blockchain network that can distribute rmware les in a Peer-to-Peer (P2P) man-ner without using a rmware server [13], [14]. The manifest le is distributed through the blockchain network and con-tains a unique hash value of the rmware le. The rmware les are stored in a distributed le system and the hash values of rmware images are stored in the blockchain. Since IoT devices have very limited computational resources, we ensure that all veri cations are performed by blockchain nodes, not IoT devices. In order to solve the bottleneck problem in the blockchain network, registration and retrieval nodes are con gured in our architecture. The following entities are de ned in the proposed architecture.

Author: An author is a manufacturer produces IoT devices or a vendor develops the rmware and the man-ifest les. An author has to upload the manifest le and the rmware image to the blockchain network to distribute.

IoT Device: An IoT device can be any IoT products operating with MCUs, sensors or actuators [10]. An IoT device needs the manifest le and the rmware image to be updated.

Registration Node: A registration node is a blockchain node that handles the author’s registration. An author should be registered to the blockchain network via regis-tration nodes to upload the manifest and rmware image les.

Retrieval Node: A retrieval node is a blockchain node that handles IoT devices downloading manifest and rmware image les via URIs contained in the manifest le.

General Node: A general node is a normal node consti-tuting the blockchain network besides registration nodes and a retrieval nodes.

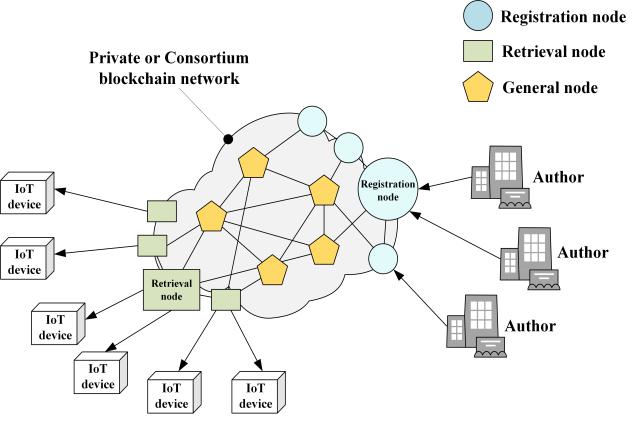
Blockchain Nodes: Blockchain nodes mean all of the nodes constituting the blockchain network, which are registration nodes, retrieval nodes, and general nodes. Blockchain nodes can be separated into two different types that are author nodes and non-author nodes. The author node is a blockchain node which is managed by the author, and the non-author node is a blockchain

node which is a voluntary or a selected node that certi ed by the author. The distribution le system for rmware images are con gured with blockchain nodes.

Every node constructing the blockchain network performs the designated role as a blockchain node. The rmware image les stored distributed on the blockchain network. The IoT devices are required to have a public key or a pre-shared key of the device’s author. Figure [3](#page4) shows the pro-posed decentralized rmware update architecture based on blockchain.

VOLUME 8, 2020

**FIGURE 3.** Proposed blockchain-based firmware update architecture.



**A. PROCEDURE DESCRIPTION**

We explain the procedure of our architecture with Figure [4.](#page5)

1) AUTHOR REGISTRATION

The manifest le contains the rmware information and a URI for downloading the rmware image which is the rmware binary of a complete software or a subset of a soft-ware. When an author creates a manifest le and a rmware image, the rst step is to request to the registration nodes for uploading the rmware les to the blockchain network. Thus, an author should create a registration request transaction.

2) REGISTRATION CHECK

A registration node checks whether the author already has been registered by the author ID and public key from the request transaction. If the author’s request is the rst time, the author ID should be registered. The blockchain nodes then validate the manifest le with the author’s public key or pre-shred key in the transaction.

3) MANIFEST STORED ON THE BLOCKCHAIN

As the manifest le gets veri ed and stored on the blockchain, the manifest le can never be deleted or tam-pered; in other words, it become irreversible. Furthermore, integrity of the manifest le itself is guaranteed. A manifest integrity is crucial as it contains the unique hash value and other information about the rmware image.

1. FIRMWARE IMAGE STORED ON THE DISTRIBUTED FILE SYSTEM

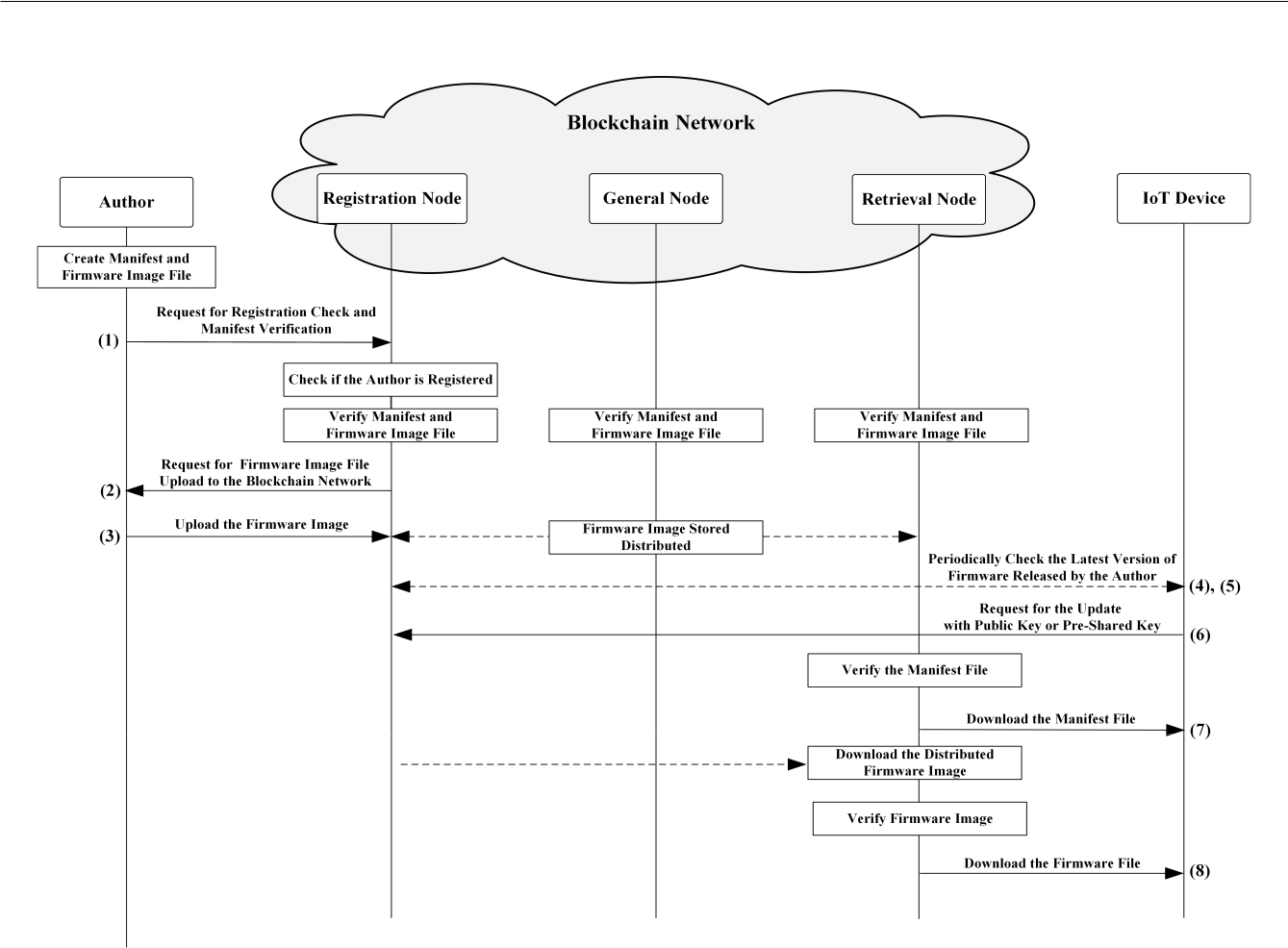
Firmware images registered and veri ed in Step (2) are stored distributed in the P2P-based le system such as IPFS [15]. The hash values of the rmware image chunks are stored on the blockchain.

1. IOT DEVICES PERIODICALLY CHECK THE FIRMWARE VERSION

The IoT devices check the blockchain network periodically. A blockchain node responds the latest rmware version

37521

S. Choi, J.-H. Lee: Blockchain-Based Distributed Firmware Update Architecture for IoT Devices



**FIGURE 4.** Procedure of the proposed blockchain-based firmware update.

released by the device’s author. Then, the IoT device com-pares the rmware version it has now, and the latest rmware version released by the author. If the rmware version that the IoT device owns is older, the IoT device requests for the latest manifest le to the retrieval node. Note that this process can be operated by a speci c device operator such as status tracker as in existing SUIT architecture [10].

6) MANIFEST DOWNLOADS

After the IoT devices request for the manifest le, the retrieval node responds to the request by sending the manifest le that was stored on the blockchain network to the requesting IoT device. We assume that the retrieval node and the IoT device have a trust relationship so that the manifest or rmware image les sent from a retrieval node to an IoT device are securely protected.

7) FIRMWARE DOWNLOADS

The retrieval node sends the manifest le to the IoT device in Step (6) and then downloads the distributed rmware image from the neighbor nodes with the information stored in the manifest. The blockchain nodes verify the downloaded rmware image with the public key or a pre-shared key of an

37522

author. The data sent from the retrieval node to the IoT device should be securely protected.

**B. TRANSACTION COMPOSITIONS**

In our architecture, 8 different types of transactions are de ned as follows (’A ! B’ implies ’sender ! recipient’, and ***Request*** or ***Response*** implies the type of the transaction):

1. Author ! Registration Node : ***Request***

An author requests for registration check and veri ca-tion of the manifest le.

1. Registration Node ! Author : ***Request***

After the manifest veri cation, Registration node requests to Author to upload the rmware image.

1. Author ! Blockchain Node : ***Response***

An author uploads the rmware image to the dis-tributed le system as a response to the request.

1. IoT Device ! Blockchain Node : ***Request***

An IoT device periodically check if the latest version of rmware is released from its author to the blockchain nodes.

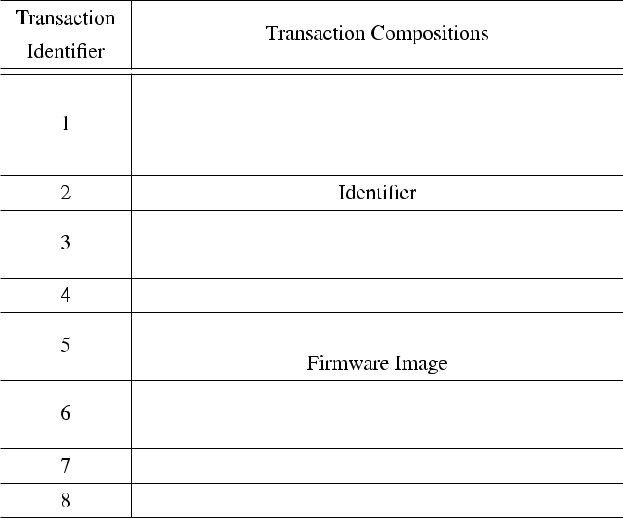
1. Blockchain Node ! IoT Device : ***Response***

A blockchain node responses to the IoT device the lat-est version of the rmware released by the IoT device’s author.

VOLUME 8, 2020

S. Choi, J.-H. Lee: Blockchain-Based Distributed Firmware Update Architecture for IoT Devices

**TABLE 2.** Compositions constituting transactions.



1. IoT Device ! Blockchain Node : ***Request***

If an IoT device con rms the rmware it has is not the latest version, it requests for an update to the blockchain nodes.

1. Retrieval Node ! IoT Device : ***Response***

A retrieval node sends the manifest le that is stored in the blockchain network to the requesting IoT device.

1. Retrieval Node ! IoT Device : ***Response***

Retrieval nodes download the distributed rmware image that is stored in the blockchain network and sends the rmware image to the requested IoT device.

Each transaction should contain its transaction type num-ber as a transaction identi er. Note that transaction can be formed in variable ways by platforms. The compositions that each transaction contains are presented in Table [2.](#page6)

**C. SAFETY MITIGATION**

Our proposed architecture is able to mitigate ’targeting issue’ and ’author-disappearing issue’ which can occured in the existing SUIT architecture. The description of each issues is as follows.

Targeting Issue: In the client-server model, a server man-ages all the data it stores, so a rmware le can be forged into a malicious le before distributed. An adversary may also delete the latest rmware le before distributed if the server is stolen. An adversary may also delete the latest rmware le. The traditional client-server struc-ture offers a highly ef cient and valued target for adver-saries since the attack on a server can damage every IoT device requesting rmware update to the server. In addi-tion, as the number of IoT devices increases exponen-tially, the rmware update requests may not be properly processed by the server due to the server overload [16]. This centralized condition provides easier environment for an adversary to conduct a network attack such as distributed denial of service attack.

Author-Disappearing Issue: Authors distributing rm-ware are not guaranteed to be permanent. Authors may disappear due to cyber-attacks or funding problems. If the author disappears before an IoT device downloads the latest version of rmware released, the IoT device will never be able to update the rmware inde nitely. This issue can lead damage of notorious cyber-attacks more serious. For example, if the latest version of rmware image contains a patch for the recent cyber attack, an attacker could make author disappeared and prevent the rmware download permanently to maintain the IoT devices vulnerable. Also, this issue means that companies with less funding power are more likely to be attacked. This issue must be resolved because it can lead to nancial polarization in terms of security and hinder

the development of the IoT rmware industry.

Both issues are single points of failure and our proposed architecture mitigate these issues with the characteristics of blockchain such as decentralization. integrity, and irre-versibility of data.

The decentralization architecture provided by the blockchain network resolves the targeting issue. Every node shares a same ledger in the blockchain network. Therefore, a targeting attack on a particular node does not act as a single point failure. Also, even if some malicious nodes send unneeded transactions repeatedly, the transactions will be distributed to the blockchain nodes by their roles. The registration nodes and retrieval nodes in our proposed archi-tecture enable ef cient distribution, and prevent bottleneck problem. In addition, if any nodes send same transactions or unveri able transactions repeatedly, a function that determine the node as a malicious node through the consensus of the blockchain nodes can be added.

If a server operated or managed by an IoT device man-ufacturer or a rmware image vendor disappears due to cyber-attacks or funding problems, all data stored on the server also become unavailable. However, the proposed archi-tecture ensures integrity and irreversibility for the hash values of the rmware image and manifest les stored on the blockchain so that provides a permanent download of rmware images even after the author is disappeared. This resolution can function differently in the consortium blockchain and the private blockchain.

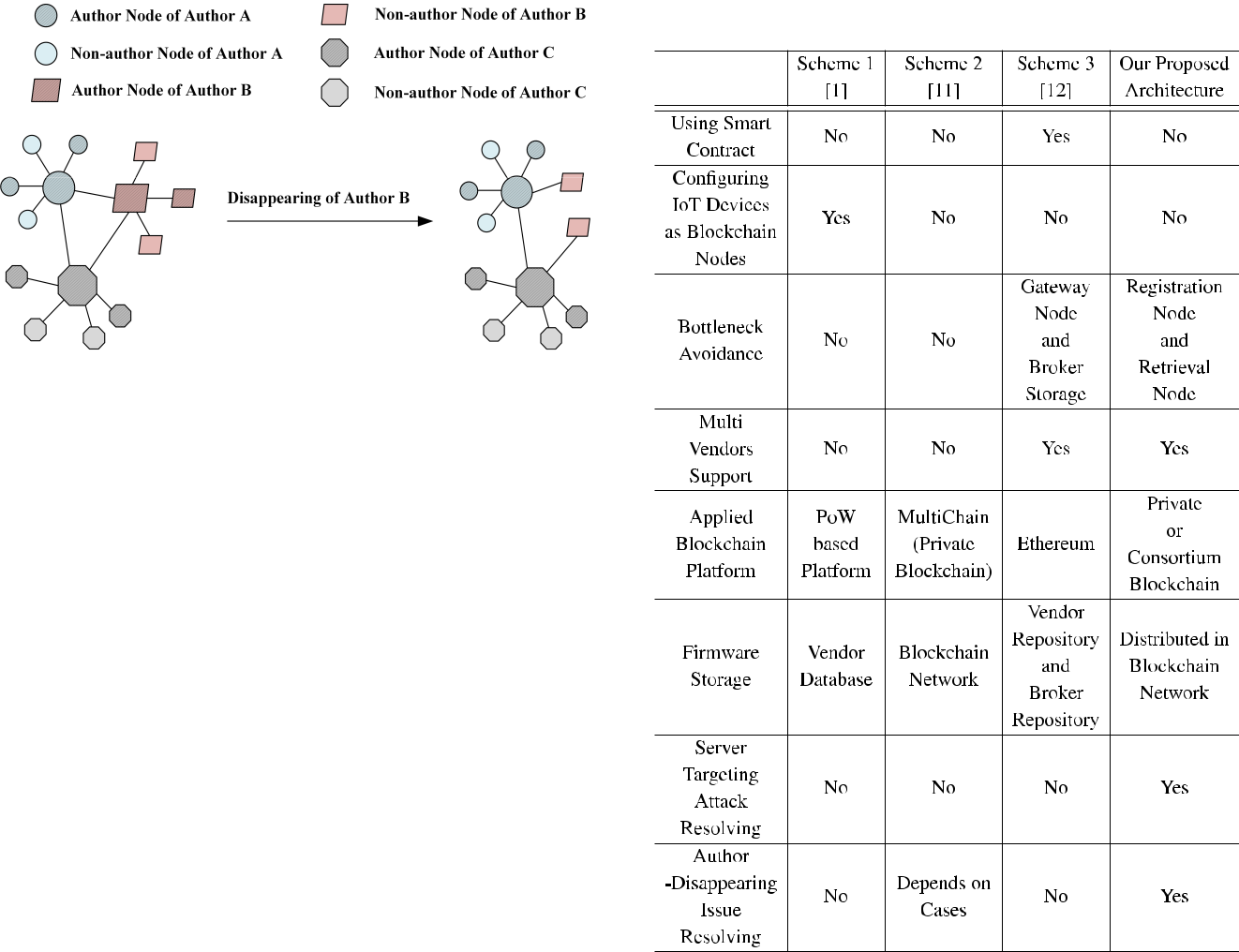
Consortium Blockchain: The consortium blockchain is operated jointly by several authors. Therefore, even if the author disappearing issue occurs in the consortium blockchain architecture, the nodes that linked to other co-authors and the non-author nodes of disappeared author retain their rmware and manifest les. The remaining nodes therefore provide irreversible down-load after the author-disappearing issue. Figure [5](#page7) shows the resolution of an author-disappearing issue in the consortium blockchain.

Private Blockchain: The private blockchain is suitable for large enterprises that are responsible for manag-ing numerous types of devices to manage and reserved

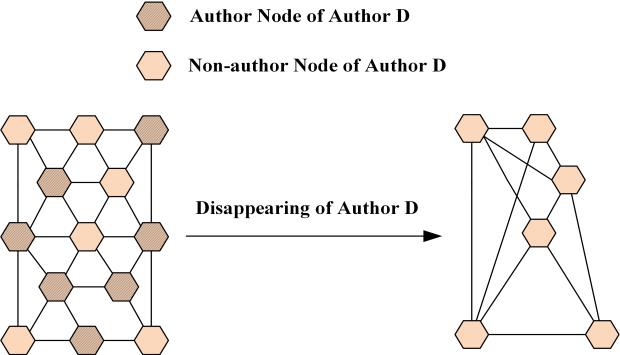
|  |  |
| --- | --- |
| VOLUME 8, 2020 | 37523 |

S. Choi, J.-H. Lee: Blockchain-Based Distributed Firmware Update Architecture for IoT Devices

**TABLE 3.** Comparison results.



**FIGURE 5.** Resolution of author-disappearing issue in the consortiumblockchain.



**FIGURE 6.** Resolution of Author-Disappearing Issue in the PrivateBlockchain.

enough credibility. In the private blockchain, if the author node are disappeared, the blockchain network only consists of non-author nodes. Therefore, a suf - cient number of non-author nodes must be guaranteed. In other words, all different types of rmware images of the author should be able to be downloaded only from the non-author nodes. Figure [6](#page7) shows the resolution when an author-disappearing issue occurs in the private blockchain.

**IV. DISCUSSION**

In this section, we provide comparison results between exist-ing blockchain-based rmware update architectures and our architecture in Table [3.](#page7)

In our proposed architecture, we do not require to apply smart contracts. However, Scheme 3 applies smart contracts that can incur excessive traf c and cost. Also, the proposed architecture does not require the IoT devices to be oper-ated as part of the blockchain network. On the other hand, Scheme 1 is consisting of the blockchain network with IoT devices even though performing veri cation and consensus algorithms as blockchain nodes require immoderate comput-ing power for IoT devices. Also, Scheme 1 applies the PoW consensus algorithm which requires too many resources.

As the number and types of IoT devices have increased, the number of rmware les that vendors have to

37524

manage, or register is also getting expanded. If a rmware le is uploaded and all corresponding devices request for downloads at once, rmware uploads from some other ven-dors would fail. Therefore, the proposed architecture con g-ures the registration node and the retrieval node to prevent this bottleneck issue.

The architecture we propose in this paper suggests both private and consortium depending on the size of the com-pany. In other words, the proposed architecture supports both single-manufacturer architecture, and multi-vendors archi-tecture. But, Scheme 1 suggests an architecture that is only suitable for a single vendor. Since the number of manu-facturer or vendors are rapidly growing, rmware update architectures should support multi-vendors.

There are various platforms in the blockchain platform, depending on the network architecture or the type of sup-porting language. However, PoW-based blockchain platforms take too much time due to very low transaction through-put and 6 con rmation [17]. Besides, blockchain platforms in public environments are more vulnerable than the per-missioned blockchain because the blockchain participants have the right to read, write data, and create blocks [18].

VOLUME 8, 2020

S. Choi, J.-H. Lee: Blockchain-Based Distributed Firmware Update Architecture for IoT Devices

The architecture proposed in this paper has a low risk of server targeting attacks because the rmware images are stored distributed on the blockchain network. However, Schemes 1, 2 and 3 store the rmware images in the cloud or storage managed by the manufacturers in centralized archi-tecture. Therefore, Scheme 1, 2 and 3 are vulnerable to server targeting attacks. Furthermore, the centralized man-agement of data is not immune to the author-disappearing issue. Scheme 2 is immune depend on cases because only the rst of the two methods proposed in Scheme 2 stores the rmware binaries in the blockchain. In our proposed architecture, the author-disappearing issue is solved through distributed storage in the blockchain network.

**V. CONCLUSION**

The IoT technology, which realizes a hyper-connected society through the interaction of things and humans, is regarded as a core technology of the 4th Industrial Revolution. How-ever, there is a lot of concern in terms of security because of its low computing power. Since the rmware update for IoT devices must be performed to the IoT devices, the secure rmware update is necessarily required. In this paper, we stated a decentralized rmware update architec-ture based on blockchain. The expressed architecture can distribute network load and guarantee integrity of rmware images. The blockchain network presented in this paper con-sists of registration nodes that process registration of manifest and rmware image les from authors, and retrieval nodes that process downloading manifest and rmware image les. The rmware les are stored in the distributed le system and the hash values of the les downloaded are stored on the blockchain. It can also resolve the server targeting and author-disappearing issues. For future work, we would like to design speci c message procedures for the transactions. Also, we look forward to implementing the architecture in both private and consortium blockchain and obtain perfor-mance evaluation and security analysis.

**REFERENCES**

1. B. Lee and J.-H. Lee, ‘‘Blockchain-based secure rmware update for embedded devices in an Internet of Things environment,’’ *J. Supercomput.*, vol. 73, no. 3, pp. 1152 1167, Sep. 2016.
2. B.-C. Choi, S.-H. Lee, J.-C. Na, and J.-H. Lee, ‘‘Secure rmware valida-tion and update for consumer devices in home networking,’’ *IEEE Trans.* *Consum. Electron.*, vol. 62, no. 1, pp. 39 44, Feb. 2016.
3. S. Nakamoto. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*.

[Online]. Available: https://bitcoin.org/bitcoin.pdf

1. J.-H. Lee, ‘‘Rise of anonymous cryptocurrencies: Brief introduction,’’ *IEEE Consum. Electron. Mag.*, vol. 8, no. 5, pp. 20 25, Sep. 2019.
2. S. Zhang and J.-H. Lee, ‘‘Smart contract-based secure model for miner reg-istration and block validation,’’ *IEEE Access*, vol. 7, pp. 132087 132094, 2019.
3. S. Zhang and J.-H. Lee, ‘‘Analysis of the main consensus protocols of blockchain,’’ *ICT Express*, to be published.
4. C. Merkle, ‘‘A Digital signature based on a conventional encryption func-tion,’’ in *Proc. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2000, pp. 369 378.

VOLUME 8, 2020

1. Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, ‘‘An overview of blockchain technology: Architecture, consensus, and future trends,’’ in *Proc. IEEE Int. Congr. Big Data (BigData Congress)*, Honolulu, HI, USA,Jun. 2017, pp. 557 564.
2. B. Moran, H. Tschofenig, and H. Birkholz, *SUIT CBOR Manifest Seri-alization*, document draft-moran-suit-manifest-05, 2019. [Online]. Avail-able: https://tools.ietf.org/html/draft-moran-suit-manifest-05
3. B. Moran, H. Tschofenig, D. Brown, and M. Meriac, *A Firmware Update* *Architecture for Internet of Things*, document draft-ietf-suit-architecture-08, 2019. [Online]. Available: https://tools.ietf.org/html/draft-ietf-suit-architecture-08
4. A. Boudguiga, N. Bouzerna, L. Granboulan, A. Olivereau, F. Quesnel, A. Roger, and R. Sirdey, ‘‘Towards better availability and accountability for IoT updates by means of a blockchain,’’ in *Proc. IEEE Eur. Symp. Secur.* *Privacy Workshops (EuroS&PW)*, Apr. 2017, pp. 50 58.
5. A. Yohan and N.-W. Lo, ‘‘An over-the-blockchain rmware update frame-work for IoT devices,’’ in *Proc. IEEE Conf. Dependable Secure Comput.* *(DSC)*, Kaohsiung, Taiwan, Dec. 2018, pp. 1 8.
6. S. Wang, Y. Zhang, and Y. Zhang, ‘‘A blockchain-based framework for data sharing with ne-grained access control in decentralized storage systems,’’ *IEEE Access*, vol. 6, pp. 38437 38450, 2018.
7. S. Muralidharan and H. Ko, ‘‘An InterPlanetary le system (IPFS) based IoT framework,’’ in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Las Vegas, NV, USA, Jan. 2019, pp. 1 2.
8. J. Benet, ‘‘IPFS content addressed, versioned, P2P le system,’’ 2014, *arXiv:1407.3561*. [Online]. Available: http://arxiv.org/abs/1407.3561
9. K. Zandberg, K. Schleiser, F. Acosta, H. Tschofenig, and E. Baccelli, ‘‘Secure rmware updates for constrained IoT devices using open stan-dards: A reality check,’’ *IEEE Access*, vol. 7, pp. 71907 71920, 2019.
10. M. Vukoli¢, ‘‘The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication,’’ in *Proc. Int. Workshop Open Problems Netw. Secur.* Cham, Switzerland: Springer, 2015.
11. S. Zhang and J.-H. Lee, ‘‘Double-spending with a sybil attack in the bitcoin decentralized network,’’ *IEEE Trans Ind. Informat.*, vol. 15, no. 10, pp. 5715 5722, Oct. 2019.

SEOYUN CHOI is currently pursuing the bach-elor’s degree in computer engineering with Sangmyung University, Cheonan, South Korea. She is also a Research Assistant with the Protocol Engineering Laboratory, Sangmyung University. Her research interests include protocol engineer-ing, network security, and blockchians.



JONG-HYOUK LEE (Senior Member, IEEE) received the Ph.D. degree in computer engi-neering from Sungkyunkwan University, Suwon, South Korea. He was with INRIA, France, for IPV6 vehicular communication and security research. He was an Assistant Professor with the TELECOM Bretagne, France. In 2013, he moved to Sangmyung University, Cheonan, South Korea. He has authored the Internet Standards: IETF RFC 8127, IETF RFC 8191, and IETF RFC 8691. His



research interests include protocol engineering and performance analysis. He was a recipient of the Best Paper Award from the IEEE WiMob 2012, the 2015 Best Land Transportation Paper Award from the IEEE Vehicular Technology Society, the Haedong Young Scholar Award, in 2017, and the IEEE SYSTEMS JOURNAL Best Paper Award from the IEEE Systems Council, in 2018.



37525