Computer Security Symposium 2019
21 - 24 October 2019

Development of a firmware authenticating and updating scheme for smart home IoT devices using distributed ledger technologies

W M A B Wijesundara[1,*], Joong-Sun Lee[1], Dara Tith[1], Hiroyuki Suzuki[1], Takashi Obi[1]

Abstract: Latest home appliances are integrated with features to assure compatibility with smart home IoT (Internet of Things). However, applying complicated security mechanisms to IoT is limited by device hardware capabilities, making them vulnerable to attacks. Such attacks have recently become frequent. With the increase of IoT devices generating large amounts of user sensitive data, improper firmware distribution harms the user's security and privacy. To address this issue, we developed a secure verification mechanism for the firmware released by the device manufacturer. We adopted IOTA decentralized ledger platform to authenticate the firmware of the end-device. The proposed firmware update scheme consists of firmware deployment and validation through IOTA's Masked Authenticate Messaging (MAM) and indirect firmware update through MQTT standard IoT protocol. The system's functionality was implemented using IOTA's MAM run on Raspberry Pi as an IoT gateway and a Wi-Fi microcontroller, which showed the effectiveness of our approach.

Keywords: Distributed ledgers, Directed acyclic graphs, IOTA, Internet of Things (IoT), Information Security

1. Introduction

Popularity of Internet of Things (IoT) is rising parallelly with steadfast improvements in wireless communication, sensors and internet technologies [1, 2]. It is expected that most equipment ranging from mobile devices and home appliances to vehicles and utility meters, will be accessible from anywhere in the world in the near future. As a result, it is estimated that the 22 billion connected things in 2018 would increase to be 50 billion by 2030. In parallel, the data volume generated by connected devices will increase from 13.6 Zettabytes (ZB) in 2018 up to 79.4 ZB in 2025 [3]. With the exponential increase of connected devices and data volume, security has become an increasing challenge especially because introducing new antivirus software to IoT devices after manufacturing stage is difficult. Instead, a device's security depends on how much or whether the manufacturer has considered about its security [4]. Prominent security challenges for internet connected IoT devices include firmware that are not up to date [5, 6], poor authentication and microchips which are not well secured [6]. In fact, the majority of IoT devices do not include in-built mechanisms for securely updating firmware. With such limitations, devices can be easily subjected to cyber-attacks [7]. Even though advances in software engineering have facilitated improvements in programming models and testing, IoT device software still require much more development for ensuring security [4]. Therefore, updating device firmware continuously and securely is essential for guaranteeing device safety [8]. Against this backdrop, researchers [4, 9 ,10, 11] have applied blockchain technology for continuous firmware updating, claiming it to be a secure method.

Blockchain technology is gaining continuous attention as an extremely secure distributed database [12]. In a distributed ledger, the database is distributed and is administered by consensus protocol which are executed on nodes of a distributed network [13, 4]. Every node in the distributed network has identical copies and only transactions which are agreed upon the consensus will be added to the ledger [9]. Therefore, any malicious activity can be easily identified [13], which can effectively enhance security of ledger information. One of the most popular applications of the blockchain technology is virtual currencies including Bitcoin. Bitcoin verifies its blocks using the proof-of-work (PoW) algorithm [14] which involves solving a cryptographic puzzle. This mechanism takes up to an hour to finalize the transaction and consumes large amounts of energy depending on the number of miners involved in the process [9]. On the other hand, Bitcoin performs four transactions per second globally, incurring a cost of one US Dollar or more. Ethereum, another blockchain based technology, also incurs 0.5 US Dollars while performing 10 transactions per second [13]. In addition, the blockchain does not support micro-transactions sufficiently and exhibits scalability and privacy issues [6]. Hence, despite strong security, low throughput, scalability issues and considerable fee associated with the blockchain is a significant burden for industrial and financial use cases. As a solution, new distributed ledgers like IOTA , Algorand, hashgraph, and Ouroboros [13] have been introduced with improved characteristics which could substitute the blockchain.

In contrast to the blockchain, directed acyclic graph (DAG) is a technology which does not employ miners to verify transactions [14, 13, 6], and DAG based technology can be effectively used to

1 Tokyo Institute of Technology
* wijesundara.w.aa@m.titech.ac.jp

build a secure, cost effective and a faster substitute for blockchain based distributed ledgers. IOTA is one such DAG based distributed ledger technology [13, 14, 6, 15]. IOTA does not possess scalability issues and fees that most traditional distributed ledgers show [6, 13, 15]. IOTA's DAG, or in other words, "The Tangle", treats each transaction like its own block rather than binding several transactions into a single block like blockchain [9, 14]. To add a new transaction to the Tangle, two previous unapproved transactions have to be approved by the new one [14, 9]. This approval process acts as the only "fee" for the new transaction. In addition, when the number of approved transactions attached to a new transaction increases, reliability of the new transaction increases alongside [9].

IOTA offers an extension called Masked Authenticated Messaging (MAM), a second layer data communication protocol, which enables publishing and receiving encrypted data streams through the Tangle as zero-value transactions [6, 13] using a MAM client who communicates with a full node in IOTA. Therefore, IoT devices capable of running MAM client can initiate an encrypted data stream to securely communicate through IOTA Tangle.

Hence, considering security, cost and latency issues associated with previous studies, we proposed an IOTA MAM based firmware updating and authentication scheme. To the extent of our knowledge, this is the first work which uses IOTA MAM for firmware updating and authentication. A proof of concept was implemented and tested against IOTA Tangle to authenticate the integrity and authenticity of IoT device firmware and keep it up-to-date using the proposed scheme. An IoT gateway was built to bridge the gap between the vendor and IoT end device by employing MAM for firmware deployment and validation. To authenticate the firmware of the end-device, secure MQTT (Message Queuing Telemetry Transport) protocol was used between IoT gateway and IoT end devices.

2. Related work

2.1 Current firmware update methods for IoT devices

Many IoT vendors frequently identify their security vulnerabilities for IoT devices and deploy patch upgrade [16]. However, most IoT devices do not include a secure built-in method to update firmware. Such a mechanism is required to ease the process of patch upgrade to prevent vulnerable exposure that leads to cyber-attacks [7]. To satisfy users' requirements, IoT vendors use 3rd party development libraries to build their firmware, which increase device firmware vulnerability [4]. Majority of IoT vendors maintain their own firmware repository with their latest firmware. The update request is made by the device if a firmware upgrade is necessary. For authentication and validation, a digital signature based authentication is used [4]. Moran et al. (2019) proposed a method to authenticate the integrity even when the firmware repository is untrusted, by encrypting the firmware image and signing using JSON and JOSE [17]. Choi et al. (2016) take the approach of partitioning the firmware into blocks and chaining them with hash values [18]. However, their approach will consume more processing power which IoT devices lack, resulting in various practical issues.

2.2 Current blockchain based firmware update methods for IoT

Lee et al. (2017) have proposed blockchain to obtain the authenticity of firmware by letting each IoT device act as a normal node connected through the blockchain network. If a normal node is requesting firmware information, it becomes a request node. A normal node also can validate its firmware and respond to request nodes. In that case, the normal node will become a response node. Verification node is hosted by the vendor for validating firmware of the normal node. While their scheme promised secure firmware verification, their systems has to go through PoW which leads to high resource utilization, latency and scalability issues. While this model has not been implemented and analyzed in practice, they also do not propose a method to verify the integrity of the firmware file [11].

Baza et al. (2018) have proposed a blockchain based firmware update scheme for autonomous vehicles which employs smart contracts. They used Attribute-based encryption (ABE) technique to set a policy about who have the rights to download and use an update using a smart contract. Their scheme promised to decrease blockchain based operation by bundling several firmware information in to one transaction using aggregate signature scheme to reduce resource consumption [19].

However, in situations where a large number of sensors are involved (ex. smart city), use of public blockchain and PoW algorithms is limited in practicality due to their resource intensive nature. This can result in decreased scalability and performance. Moreover, the authors of both cases given above do not clearly mention how IoT devices can directly interact with blockchain with their limited computational power. Therefore, it is important to design a scalable, high performance and lightweight mechanism to ensure IoT device security. Our solution resolves this issue by using IOTA distributed ledger technology with ISO standard MQTT protocol.

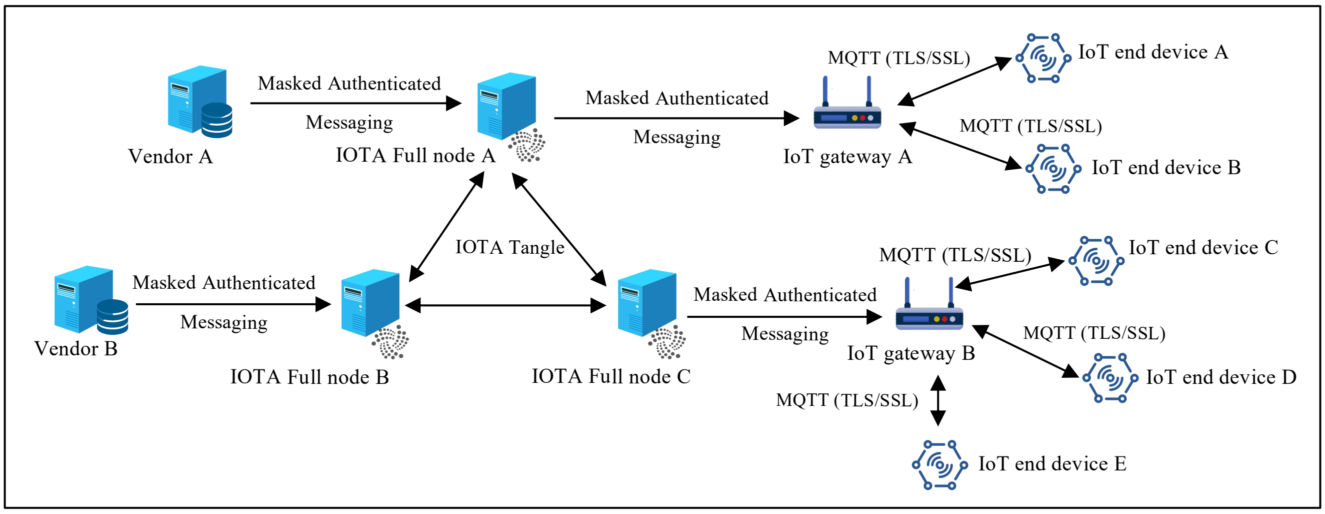


Figure 1 Overall network architecture of the proposed system

3. Proposed firmware authenticating and updating scheme

The overall network architecture of the proposed scheme is shown in Figure 1. The purpose of the proposed scheme was to establish efficient and secure communication between the vendor and the IoT end device via an IoT gateway. Hardware limitations of IoT end devices restrict their integration with IOTA MAM. Therefore, MQTT protocol was used for communication between end devices and the gateway while IOTA MAM was used for communication between the gateway and the vendor. IOTA MAM messages are communicated through the IOTA Tangle. The Tangle consists of two kinds of nodes. While full nodes download the entire DAG, a light node downloads part of the DAG. In our system, Vendors and IoT gateways communicate with MAM protocol by connecting to the Tangle through full nodes. In this section, we present details of our IOTA's MAM-based firmware updating platform. The proposed system includes following phases,

- Phase 1 - Vendor (manufacturer) and IoT gateway registration (prerequisite)
- Phase 2 - IoT end device registration with IoT gateway (prerequisite)
- Phase 3 - New firmware update distribution by the vendor
- Phase 4 - IoT gateways receiving firmware information from vendors
- Phase 5 - Verification and methods of firmware updating by IoT gateway and IoT end devices

There are three entities in the proposed firmware update scheme whose functions are further described in Table 1. Each phase of the firmware updating platform is described further hereafter.

Phase 1 - Vendor (manufacturer) and IoT gateway registration

To publish messages to IOTA Tangle as a MAM message, the publisher uses a channel ID. Anyone can receive that message from IOTA Tangle using the channel ID.

Table 1 Entities of the proposed firmware update scheme

Entity	Description
Vendor	● A firmware repository is owned by the device manufacturer to store firmware binaries and firmware information ● MAM client is installed in vendor's node to initiate a MAM connection with IoT gateway ● Each vendor has their own root and side key for their restricted MAM channel
IoT gateway	● A gateway with Wi-Fi, which is connected to the vendor's node via IOTA MAM ● Stores information about the connected IoT end device's device type, device MAC address, current firmware version, and current firmware hash ● MAM client is installed in IoT gateway to initiate a MAM connection with vendor node ● Vendors Root and side key is required in advance to establish the MAM connection ● MQTT server is running on IoT gateway; every IoT end device is connected to IoT gateway via MQTT connection over Secure Socket Layer (SSL)/Transport Layer Security (TLS) ● Consists of Network Time Protocol (NTP) service for providing date and time for IoT end devices ● There is a separate web interface running on IoT gateway for configuring and monitoring the current status of MAM connection and MQTT connection with IoT end devices
IoT end device	● The Smart Home IoT device ● Runs MQTT client over SSL/TLS connected to the IoT gateway over Wi-Fi

MAM uses a Merkle tree-based signature scheme for message encryption and uses three types of privacy control modes. In public mode, the Merkle Tree's root is the address of the transaction. Receiver can decrypt MAM message using address/root. In private mode, the hash of the Merkle Tree's root is used as the transaction address. Merkle Tree's root is required for MAM decryption. The restricted mode uses encryption features of private mode and a side key to increase security. In order to decrypt MAM messages, receiver needs Merkle Tree's root and side key [20, 13].

We adopted restricted mode of MAM to establish the secured connection between the vendor and IoT gateway. If a vendor would like to grant access to their firmware information feed, the vendor and IoT gateway must share MAM root ID and side key in advance to initiate a restricted MAM connection. In return an IoT gateway retrieves and authenticates the associated data stream from the Tangle. When IoT devices from different vendors (i.e. different brands) are connected to the IoT gateway, the gateway has to subscribe to each vendor's MAM channel using their authorization keys to receive firmware updates. If the vendor would like to revoke access to their data streams at any time, they could update their MAM channel's authorization key, after which corresponding subscribers will not be able to decrypt MAM messages.

Phase 2 - IoT end device registration with IoT gateway

Every IoT end device has to connect to its IoT gateway via Wi-Fi. Therefore, Wi-Fi credentials, MQTT credentials and SSL public key must be shared in advance to initiate a secure connection between IoT gateway and IoT end device. With valid credentials, IoT end device will connect to IoT gateway and synchronize date and time. Then it will listen to the IoT gateway for any firmware updates information through MQTT protocol. IoT gateway will uniquely identify each IoT end device with their MAC address.

Phase 3 – New firmware update distribution by the vendor (Figure 2)

Every vendor node consists of a web interface that runs on Node.js with MAM client. Firmware deployment is carried out through this web interface. This web portal is capable of attaching MAM messages to the IOTA Tangle. When there is a new firmware update to be deployed by the vendor, authorized vendor personnel will access the vendor portal and upload the firmware binaries along with firmware information. The portal

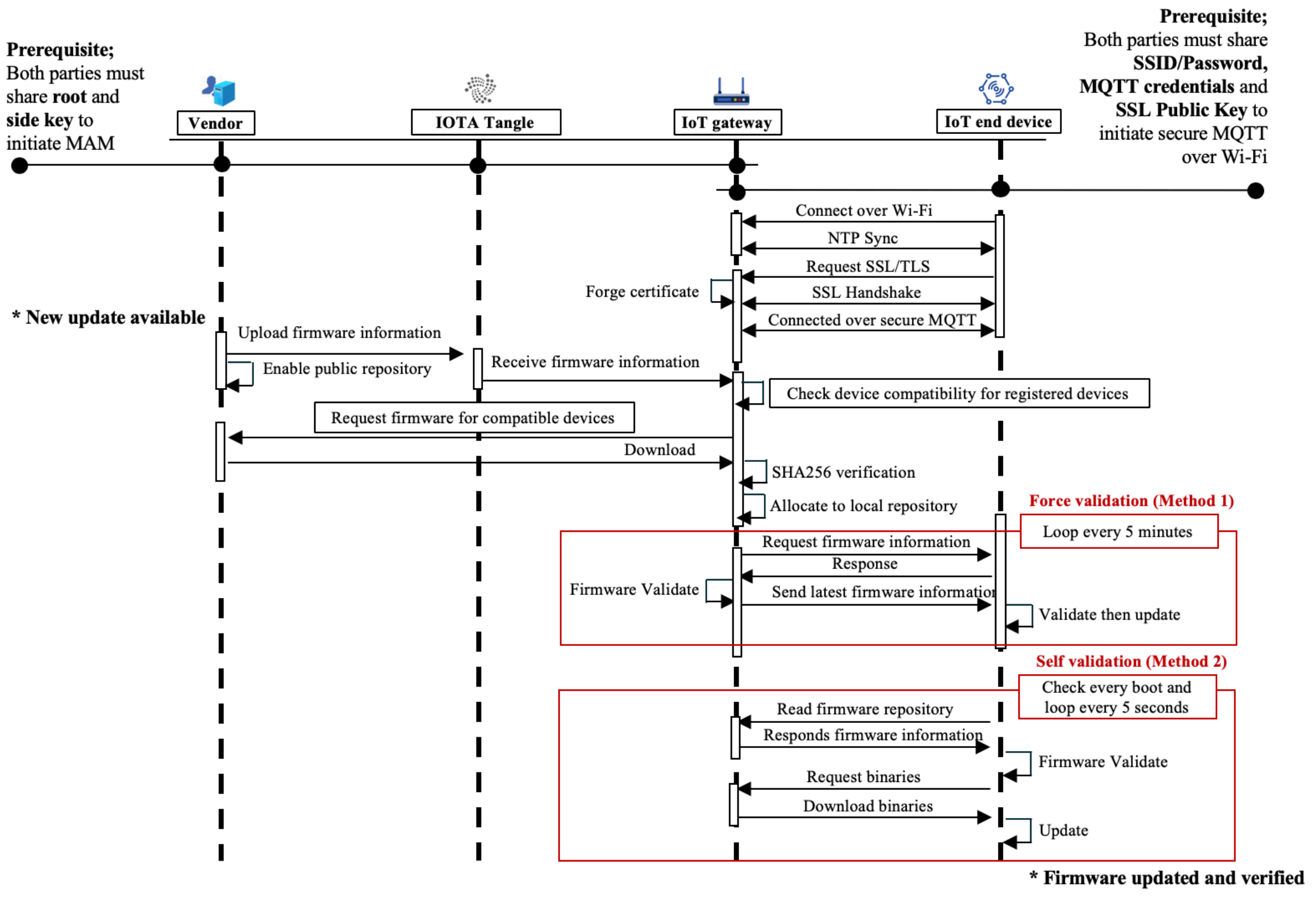

Prerequisite;
Both parties must share root and side key to initiate MAM
Vendor
IOTA Tangle
IoT gateway
IoT end device
Prerequisite;
Both parties must share SSID/Password, MQTT credentials and SSL Public Key to initiate secure MQTT over Wi-Fi
Connect over Wi-Fi
NTP Sync
Request SSL/TLS
Forge certificate
SSL Handshake
Connected over secure MQTT
* New update available
Upload firmware information
Enable public repository
Receive firmware information
Check device compatibility for registered devices
Request firmware for compatible devices
Download
SHA256 verification
Allocate to local repository
Force validation (Method 1)
Loop every 5 minutes
Request firmware information
Response
Firmware Validate
Send latest firmware information
Validate then update
Self validation (Method 2)
Check every boot and loop every 5 seconds
Read firmware repository
Responds firmware information
Firmware Validate
Request binaries
Download binaries
Update
* Firmware updated and verified

Figure 2 Firmware deployment and updating

then calculates the corresponding SHA256 value for the binary file uploaded, and generates a public URL to access it via the network. Once the vendor approves to publish information to the Tangle, the portal will include date and time, file size, file hash (sha256), firmware version, compatible hardware version, device model/type and a public URL to a MAM message through restricted channel. Upload status will be notified to the vendor through the web portal.

Phase 4 - IoT gateways receiving firmware information from vendors (Figure 2)

Each IoT gateway is connected to vendor/s with their Root ID and side key and continuously listens to the vendor's update through IOTA Tangle via MAM. This process runs every five minutes to reduce processing power and network traffic. When there is new information received through the Tangle, it will crosscheck firmware device compatibility with the currently registered IoT end devices.

Phase 5 - Verification and methods of firmware updating by IoT gateway and IoT end devices (Figure 2)

The IoT gateway will receive firmware updates corresponding to the registered IoT end devices. When a new firmware update information is received from the vendor, it will check the compatibility of latest firmware information with registered devices. If the results came out verified, it will download the latest firmware to a temporary local location and run sha256 verification for the downloaded binaries. If the hash verification is successful, it will move firmware files to the local repository and push the latest firmware information to IoT devices. All the IoT end devices which are connected (online) to the IoT gateway at the time, will receive a message for immediate firmware update. This process is called force validation (Method 1).

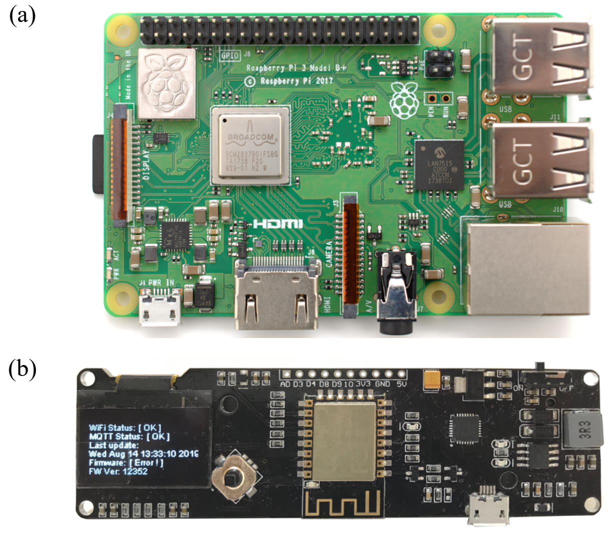
IoT end devices which are offline or not connected will be able to receive new firmware information once the device is registered with IoT gateway and become online. Every IoT end device is configured to self-check latest firmware information from IoT gateway every five seconds. This process is called self-validation (Method 2). IoT devices will access the IoT gateway's firmware repository to check the latest firmware information. If the IoT device detects new firmware, it will run a hash verification. When the firmware binaries are authenticated, it will upgrade its firmware and then reboot with the latest.

4. Implementation

We built a concept-proof prototype using IOTA MAM running on two Linux Raspberry Pi 3B+ (Figure 3.a) as IoT gateway and Vendor's node. As the IoT end device, we used ESP8266 Wi-Fi chip with OLED display (Figure 3.b). We used IOTA live network with our prototype with two full nodes configured in Tokyo Institute of Technology, Suzukakedai campus (Intel(R) Core(TM) i3-5010U CPU @ 2.10GHz with 8GB RAM running Ubuntu 18.04 LTS) and Ookayama (Intel(R) Core(TM) i7-8650U CPU @ 1.90GHz with 32GB RAM running Ubuntu 18.04 LTS). IoT gateway implementation included two daemon services written in Node.js.

- Communicating with vendors through IOTA MAM, verifying the firmware information, converting that information to MQTT messages and sending it to IoT end devices
- Web interface for administration and debugging the system

IoT end device was written in C++ with HTTPS client MQTT libraries. The 128x64 OLED display was used with IoT end device for debugging purposes. IoT end devices communicated with the IoT gateway using MQTT protocol through SSL/TLS.



```
WiFi Status: [ OK ]
MQTT Status: [ OK ]
Last update:
Wed Aug 14 13:33:10 201
Firmware: [ Error ! ]
FW Ver: 12352
```

Figure 3 (a) Linux Raspberry Pi 3B+ and (b) ESP8266 Wi-Fi chip with OLED display

The vendor node and IoT gateway communicated with each other using IOTA MAM client run on Node.js. These MAM clients directly communicated to IOTA Tangle through IOTA full node. When there is a new firmware update available, authorized person from manufacture needs to access the vendor's interface and upload the .bin file with the firmware information. After that, the web portal calculates the hash value for .bin file and store that file in a public location by generating a public URL. Along with firmware information, verification values and public URL will attach to the IOTA tangle as a MAM message. All the IoT gateways subscribed to vendors firmware feed in advance receive this information through IOTA MAM client. After decrypting the message received, it runs the verification methods previously explained and downloads the firmware and stores in a local repository. Our implementation source code is available at http://github.anushkawijesundara.com

5. Results and discussion

We evaluated the proposed system performance in terms of authenticity, scalability, latency and transactions fees. In current standard firmware update methods used for many IoT devices, some vendors provide validation information with their repositories, such as hash values to authenticate their binaries. However, that process is not autonomous and scalable. Compared to the blockchain based firmware update method by Lee et al (2017) [11], our method performs a hash verification for firmware binaries and autonomous firmware update mechanism to send a message through IOTA tangle. Also, our method is highly scalable, low latency and no transaction fees are applied since we use IOTA protocol instead of blockchain technology to avoid latency, fees and transaction bottlenecks (Table 2).

Table 2 Functionality comparison

Feature	Current firmware update methods	Blockchain based firmware update method (Lee et al 2017)	Our firmware update method
Secure firmware validation	Vendor dependent	Yes	Yes
Firmware integrity authentication	No	No	Yes
Autonomous firmware update	No	No	Yes
Transaction fees	NA	Yes	No
Scalability	Low	Average	High
Message latency	NA	High	Low

6. Conclusion

In this paper we proposed an IoT gateway for secure firmware verification and updating for smart home IoT devices utilizing IOTA MAM protocol. The proposed scheme ensures the authenticity and firmware integrity of IoT end devices. Two types of validation methods were proposed for firmware updating and validation. We implemented the proposed scheme using Node.js with IOTA MAM client libraries along with a web interface to configure and monitor the connectivity between IOTA MAM and IoT end devices. Our system gave promising results of performing secure automated firmware updates on IoT end devices with very low computational power, such as ESP8266 Wi-Fi microcontroller and low message latency through IOTA. To avoid single point of failure in sharing firmware files, our future work will focus on how to inherit IPFS distributed file sharing technology to transfer firmware files from the vendor to IoT gateway.

References:

[1] Q. Xu, K. Mi, M. Aung, Y. Zhu, and K. L. Yong, "A Blockchain-Based Storage System for Data Analytics in the Internet of Things," *Springer Int. Publ. AG 2018*, vol. 715, pp. 119–138, 2018.

[2] L. Chen *et al.*, "Robustness, Security and Privacy in Location-Based Services for Future IoT: A Survey," *IEEE Access*, vol. 5, pp. 8956–8977, 2017.

[3] S. Liu, "IoT connected devices worldwide 2030," 2019. [Online]. Available: https://www.statista.com/study/27915/internet-of-things-iot-statista-dossier/.

[4] J.-W. Hu, L.-Y. Yeh, S.-W. Liao, and C.-S. Yang, "Autonomous and Malware-proof Blockchain-based Firmware Update Platform with Efficient Batch Verification for Internet of Things Devices," *Comput. Secur.*, vol. 86, no. 2019, pp. 238–252, 2019.

[5] Symantec, "Internet Security Threat Report," 2019.

[6] U. Sarfraz, M. Alam, S. Zeadally, and A. Khan, "Privacy aware IOTA ledger: Decentralized mixing and unlinkable IOTA transactions," *Comput. Networks*, vol. 148, pp. 361–372, Jan. 2019.

[7] K. Zandberg, K. Schleiser, F. Acosta, H. Tschofenig, and E. Baccelli, "Secure Firmware Updates for Constrained IoT Devices Using Open Standards: A Reality Check," *IEEE Access*, vol. 7, pp. 71907–71920, 2019.

[8] Y. Zhao, Y. Liu, A. Tian, Y. Yu, and X. Du, "Blockchain based privacy-preserving software updates with proof-of-delivery for Internet of Things," *J. Parallel Distrib. Comput.*, vol. 132, pp. 141–149, 2019.

[9] A. Yohan, N. Lo, and S. Achawapong, "Blockchain-based Firmware Update Framework for Internet-of-Things Environment," *Conf. Inf. Knowl. Eng.*, pp. 151–155, 2018.

[10] R. of K. Jea-Min Lim (College of Informatics Korea University Seoul, Republic of Korea), Youngpil Kim (College of Informatics Korea University Seoul, Republic of Korea), Chuck Yoo (College of Informatics Korea University Seoul, "ChainVeri: Blockchain-based Firmware Verification System for IoT environment," 2018, pp. 1050–1056.

[11] B. Lee and J. H. Lee, "Blockchain-based secure firmware update for embedded devices in an Internet of Things environment," *J. Supercomput.*, vol. 73, no. 3, pp. 1152–1167, 2017.

[12] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf.

[13] J. Brogan, I. Baskaran, and N. Ramachandran, “Authenticating Health Activity Data Using Distributed Ledger Technologies,” *Comput. Struct. Biotechnol. J.*, vol. 16, pp. 257–266, Jan. 2018.

[14] A. Raschendorfer *et al.*, “On IOTA as a potential enabler for an M2M economy in manufacturing,” *Procedia CIRP*, vol. 79, pp. 379–384, 2019.

[15] S. Popov, O. Saa, and P. Finardi, “Equilibria in the Tangle,” *Comput. Ind. Eng.*, vol. 136, no. July, pp. 160–172, 2017.

[16] M. P. Xinchi He, Sarra Alqahtani, Rose Gamble, “Securing Over–The–Air IoT Firmware Updates using Blockchain,” 2019, pp. 1–23.

[17] D. B. (Linaro) Moran, B. (Arm Limited), M. Meriac (ConsultantTschofenig, H. Arm Limited), “A Firmware Update Architecture for Internet of Things Devices draft-ietf-suit-architecture-05.”

[18] B. C. Choi, S. H. Lee, J. C. Na, and J. H. Lee, “Secure firmware validation and update for consumer devices in home networking,” *IEEE Trans. Consum. Electron.*, vol. 62, no. 1, pp. 39–44, 2016.

[19] M. Baza, M. Nabil, N. Lasla, K. Fidan, M. Mahmoud, and M. Abdallah, “Blockchain-based Firmware Update Scheme Tailored for Autonomous Vehicles,” Nov. 2018.

[20] I. Sandeep, Kiran Pinjala(Dept. of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India), Krishna, M. Sivalingam (Dept. of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, “DCACI: A Decentralized Lightweight Capability Based Access Control Framework using IOTA for Internet of Things,” 2019.