

Research article

Blockchain-assisted authenticated key agreement scheme for IoT-based healthcare system

Ashish Tomar*, Niraj Gupta, Divya Rani, Sachin Tripathi

Indian Institute of Technology (Indian School of Mines), Dhanbad, Dhanbad, India



ARTICLE INFO

Keywords:

Internet of Medical Things
Authenticated key exchange
Blockchain
Security
Privacy

ABSTRACT

IoT devices are widely used in smart hospitals, digital health, and automated pathology laboratories to generate a large amount of heterogeneous data. These devices, coupled with medical sensors, enable healthcare professionals to monitor patients' health and medical device performance in real-time. The collected health data is transmitted to centralized cloud servers for analysis and diagnosis. However, the wireless communication channels used by these devices are vulnerable to security threats such as unauthorized access, denial-of-service attacks, and man-in-the-middle attacks. Existing solutions rely on a single trusted authority for computing and storage, which can lead to issues such as higher latency, centralization, and a single point of failure. To address these concerns, we propose a new protocol called Blockchain-based IoMT Authenticated Key Exchange (BioMTAKE) that establishes a private/consortium blockchain-based distributed environment using Hyperledger Fabric. This protocol eliminates the need for a single trusted authority and ensures secure access to data generated by IoMT devices. Before sharing or accessing any data from the distributed healthcare system, the BioMTAKE scheme establishes a secured shared session for authenticated devices to prevent unauthorized access. The proposed methodology has been subjected to formal security analysis using the Scyther tool and informal security analysis, which demonstrates its strong security. Additionally, our performance analysis using Hyperledger Fabric and cryptography libraries shows that the proposed scheme is computationally and communicatively efficient.

1. Introduction

The connectivity of smart devices and sensors has become mainstream since the introduction of Internet of Things (IoT) technologies. These devices have both processing and communication capabilities, allowing them to share and process data with different devices [1]. The use of paper-based medical information has frequently affected traditional healthcare methods, which have now transitioned into electronic patient records. For business modernization, IoT empowers healthcare organizations with new capabilities and opportunities but also introduces various challenges and security risks. The Internet of Medical Things devices (IoMT) is an important IoT category that can be utilized to gather and analyze health information from patients, resulting in a better quality of life along with lower medical costs. Patients, smart equipment, healthcare workers, clinical gadgets, robots, and an infinite number of wireless sensors are all part of this ecosystem, and they may all exchange critical data [2].

Medical data generated are extremely important and must be handled with caution to avoid data manipulation. Unfortunately, such personal information has been the subject of a number of cyber-attacks. Numerous medical organizations have always been hacked to date, with millions of patient records compromised [3]. Many regulations have been enacted to give healthcare

* Corresponding author.

E-mail address: ashishtomar244@gmail.com (A. Tomar).

<https://doi.org/10.1016/j.iot.2023.100849>

Received 10 January 2023; Received in revised form 15 June 2023; Accepted 15 June 2023

Available online 26 June 2023

2542-6605/© 2023 Elsevier B.V. All rights reserved.

systems instructions on how user information must be processed, maintained, and secured to prevent theft and fraud. Amidst this, hackers appear to find the healthcare business to be an easy target, owing to the absence of technological awareness inside the field. Continuous ongoing attacks on healthcare businesses demonstrate the sector's data privacy difficulties. Phishing attacks and ransomware are examples of target attacks that are successful multiple times in retrieving personal information. Indeed, many successful ransomware attacks on these institutions demonstrate the absence of simple security and privacy measures like system updates and backups. Authentication is a major security requirement where any user or device is verified on the basis of some predefined security parameters before accessing the network. Furthermore, the key agreement process establishes a common key between participating entities to enable secure data transfer. [4,5].

Blockchain has established itself as the most secure and decentralized platform available. It has numerous significant properties like tamper-proofing, traceability, immutability, confidentiality, data integrity, and privacy without using a third party. Many researchers have discovered the blockchain application as an effective security measure for the healthcare system. When it comes to blockchain significance in the healthcare environment, it is about more than just the standard transparency and security concerns. According to IBM, 70% of healthcare leaders predict that the greatest impact of blockchain within the health domain will be the improvement of clinical trial management, regulatory compliance, and providing a decentralized framework for sharing electronic health records (EHR). In short, blockchain guarantees data integrity, and data ownership remains with consumers rather than healthcare organizations. Recently, using blockchain technology, numerous academics have explored and implemented security methods in the context of cloud-fog-based IoMT [6–9].

The goal of this work is to conduct a critical analysis of factors affecting design of authentication scheme for healthcare environment and blockchain integration to design a distributed healthcare architecture for the smooth and highly secure functioning of the system.

1.1. Motivation and contributions

The COVID-19 outbreak has reshaped the way frontline workers, particularly health professionals, operate. IoT-based telemedicine techniques and digital registers are being largely replaced in place of in-person patient monitoring and record keeping. Due to a lack of technological awareness inside the healthcare field, the deployed IoT networks use vulnerable wireless communication mediums and lack sturdy security schemes for resource-constrained systems, which leads to the network being more susceptible to cyber-attacks. The Cyber-attacks on these networks might cause property damage to hospitals or risk the lives of patients. Accordingly, the data must be processed, maintained, and secured to prevent theft and fraud. Mutual authentication and key agreement are regarded as strong solutions for dealing with security weaknesses such as security attacks and privacy issues. Thus, implementing mutual authentication within the network along with key agreement will enable authenticated and secure data transfer among healthcare network entities. Therefore, numerous mutual authentication and key agreement schemes have been presented in the literature for the healthcare environment.

The common problem that exists with the currently existing schemes is their over-reliance on a single trusted authority. Various issues arise due to the centralized architecture, including single points of failure and centralized data storage. Also, the existing schemes suffer from scalability issues. As a result, we proposed the following approach to solve the issues of over-reliance on a single trusted authority, scalability, security weaknesses, and high storage and transmission costs:

1. We propose a distributed healthcare architecture that integrates Hyperledger fabric blockchain in the healthcare environment by utilizing fog servers as blockchain peers that enables scalability, flexibility, high degrees of integrity, and resiliency.
2. We also designed a mutual authentication and key agreement protocol that enables mutual authentication and the generation of shared keys in order to establish trust and secure communication between data collector nodes and IoMT devices.
3. Next, the proposed BioMTAKE scheme is proven secure using informal security analysis and formal verification based on the Scyther tool.
4. At last, we have used Hyperledger Fabric version 2.1 for implementing blockchain operations and the Cryptography library for cryptographic operation evaluations. In the performance analysis, we have compared our BioMTAKE scheme with other existing schemes.

The rest of the paper has been organized as follows: Section 2 describes the state-of-the-art research related to our work. Next, in Section 3, we gave some details about preliminary knowledge and the network model of the proposed scheme. Section 4 describes our proposed scheme, and Section 5 illustrates the security of the proposed scheme formally and informally. Finally, in Section 6, we demonstrate performance analysis of the proposed scheme, and in Section 7, we have concluded our scheme.

2. Related work

Many authentications and key agreement approaches have developed in recent years to address privacy and security issues in healthcare systems integrating IoT devices. For example, in paper [10], Yeng et al. focused on establishing security awareness in healthcare employees and delivered a thorough examination of healthcare security. There are two types of healthcare application architectures: (1) centralized solutions and (2) distributed solutions. Several centralized designs have been investigated in the healthcare business, especially for handling EHRs and patient data. For wireless body area networks in the healthcare IoT, Fotouhi et al. [11] developed a two-factor, simple authentication mechanism. The suggested protocol can be safe against several known

attacks and offer forward secure protection as it is based on the hash chain. Later, Li et al. [12] presented PSL-MAAKA, which was a lightweight mutually authenticated and key agreement protocol for wireless medical sensor networks that uses XOR and hash operations across completely public channels. Masud et al. [13] proposed a protocol that tackles security risks, by using a user authentication system for healthcare-related IoT that is privacy-preserving, efficient and lightweight. Their suggested technique is based only on lightweight bitwise XoR operations and hash function that efficiently decreases sensor node pressure. These primitives' operations might be used to develop an extremely lightweight authentication protocol at the sacrifice of some security. The main drawback of the method was that it relied on a single Trusted node for all of its authentication and storage of the data. Since the server's capacity is restricted and cannot sustain infinite traffic, centralized solutions have scalability issues. There is the problem of a single point of failure due to this configuration. Amintoosi et al. [14] proposed a lightweight scheme to achieve authentication in IoMT-based healthcare applications. They utilized a Scyther tool to prove the security strength of their proposed scheme. Thus, Decentralization can be achieved through blockchain which also encompasses all the security features in it [15]. For WMSN, Wang et al. [7] suggested a lightweight and secure authentication mechanism based on physically unclonable features (PUF) and cutting-edge blockchain technology. This took advantage of blockchain technology as well as the lightweight algorithm for the key agreement mutual authentication. However, their scheme is proven insecure by Yu and Park [16]. They proved attacks such as man-in-the-middle and session key disclosure and also showed incorrectness in their mutual authentication process. Recently, Jia et al. [17] proposed a fog computing-based general architecture for the IoMT environment. They proposed an authentication scheme using ECC between the end user and the fog node. They used blockchain to check the authenticity of the public keys. They also proposed a PUF-based scheme to provide authentication between the smart device and the fog node. Also, Griggs et al. [8] used Ethereum-based public blockchain to maintain secure records. Even though there is a lot of advantage of public blockchain we cannot rely on a public blockchain for healthcare systems as the data needs to be regulated and secure. For this, we must use private or consortium-based blockchain networks. In a similar attempt to achieve secure communication, Tomar et al. [6] incorporates consortium blockchain technology for smart grids using fog and cloud nodes and three-party authentication and key agreement scheme. As mentioned, nevertheless many research papers have proposed authentication and blockchain-based healthcare systems that deal with some potential assault, centralization, scalability, and a single failure point. Thus, in this paper, we use Hyperledger Fabric, a consortium blockchain technology, and a unique lightweight authentication mechanism for healthcare systems we try to address the issues.

3. Preliminaries

In this section, we discuss the background knowledge for the proposed scheme.

3.1. Blockchain technology

At the heart of a blockchain network is a distributed ledger that records all the transactions that take place on the network. It uses cryptographic hashing and decentralization to achieve immutability and transparency among the connected blocks of transactions. To support the consistent update of information and to enable a whole host of ledger functions (transacting, querying, etc.)—a blockchain network uses chaincode to provide controlled access to the ledger. On the basis of access requirements, blockchains can be categorized into public, private, and consortium blockchains. We use the Hyperledger Fabric consortium open-source blockchain platform in our operations. It has the following components [18]:

1. **Peers:** The ledger and smart contract are held by peers who participate in the consensus process. As a result, they constitute the fundamental building component of a fabric network.
2. **Ledger:** It has two distinct, though related, parts—a blockchain and a world state, where blockchain consists of a transaction log stored in an immutable chain of data blocks, and the world state holds the current values of records in the form of key-value pairs. The ledger is distributed between participating peers.
3. **Orderer:** In the fabric network, the orderer node carries out the tasks such as transaction orders, creating blocks and distributing them to participating nodes.
4. **Chaincode:** The smart contract contains the logic for transactions in the form of executable code. Multiple smart contracts can be grouped inside a chaincode, which is later deployed on peers in a channel.
5. **Channel:** A channel is a medium of communication between participants of the consortium. A fabric network can have multiple channels, and a peer can be part of various channels.

As Hyperledger Fabric considers developers' needs for a complete toolset that can swiftly implement a variety of privacy and security standards, Hyperledger is an excellent fit for healthcare applications. Additionally, it has comprehensive control over smart contracts that may be run in a variety of computer languages, including Golang, node.js, and JavaScript. While Bitcoin and Ethereum can complete 7 and 15 transactions per second (TPS), respectively, Hyperledger outperforms the competition with a transaction speed of 3000 TPS. It does not use cryptocurrencies as a motivator. Another advantage is that it has a high transaction throughput and a cheap transaction cost.

The proposed distributed healthcare infrastructure is built on blockchain. We propose consortium Hyperledger Fabric 2.1 based blockchain architecture with Practical Byzantine Fault Tolerance (PBFT) consensus. The functionality and use of Hyperledger is to register, enroll and authenticate new nodes into the network with the help of Certificate Authority (CA). It is also used to add new organizations and peers into the network to scale up the network when required.

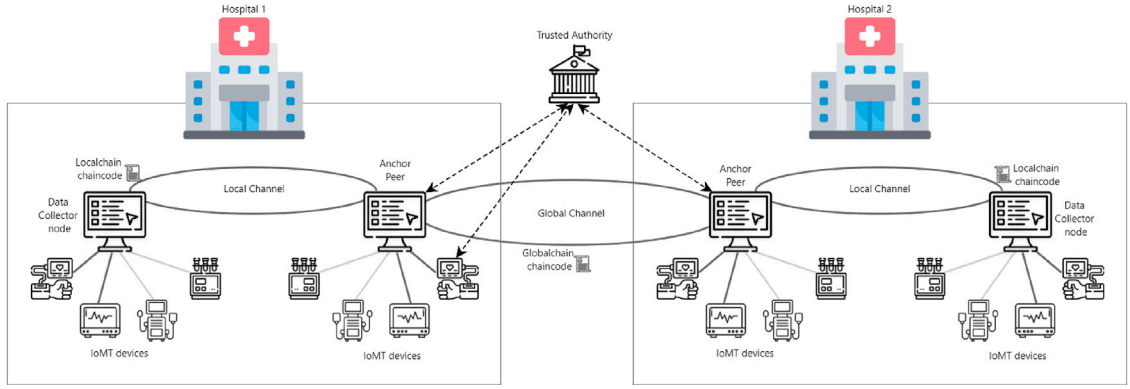


Fig. 1. Our proposed blockchain-based system model.

Notation	Description
$E_p(a,b), P$	Elliptic Curve with generator P
SK_{TA}, Pb_{TA}	Private and Public key of TA
SK_{DC}, Pb_{DC}	Private and Public key of DC
SK_{IM}, Pb_{IM}	Private and Public key of IoMT device
ID_{DC}	Real-Identity of DC
ID_{IM}, PID_{IM}	Real-Identity and Pseudo identity of IoMT device
SP	Secret Parameter
SK_{DC-IM}	Shared secret key of DC and IoMT
PT_{DC}, PT_z	Plaintext to be encrypted by ASCON encryption at DC
N_{DC}, N_x, N_z, N_p	Nonces used during ASCON encryption/decryption process
AD_z, AD_p	Associative Data used during ASCON encryption/decryption process
TM_x, TM_z	Timestamps used during authentication phase
$\Delta T, TM^*$	Allowed time delay, Received time of a message
(CT_{DC}, Tag_{DC})	Ciphertext and Tag stored at globalchain
(CT_x, Tag_x)	Ciphertext and Tag generated by IoMT device during Authentication
(CT_z, Tag_z)	Ciphertext and Tag generated by DC during Authentication
$\epsilon_K\{PT\}, D_K\{CT\}$	ASCON encryption and decryption using key K
S_x, S_z	Random number used in Authentication phase
$\oplus, \cdot, , h()$	XOR, ECC operation, Concatenation, hash function respectively

Fig. 2. List of notation used in BioMTAKE.

3.2. Network model

The network model depicted in Fig. 1 is used in this paper. According to the network model, there are Trusted Authority (TA), Data Collector Nodes/Peers (DC), and IoMT devices.

In the healthcare domain, hospitals are one of the multiple organizations involved. In this paper, we focus on two hospitals that utilize a network consisting of two channels, each with a separate ledger. The first channel is an intra-hospital channel that stores local device authentication information and data generated by medical devices until the discharge period. The second channel is related to the global blockchain maintained by multiple hospitals to store patients' records permanently. Each channel can have more than one chaincode; in our model, we have three chaincodes, each with a different endorsement policy. Chaincodes must be

installed and instantiated in every peer in the channel, and a policy must be established when a chaincode is created. In our model, each organization (hospital) has multiple peers (data collector nodes), with two in each organization. One of the peers is an anchor peer, which is discoverable to all other peers in a channel. It is used by peers to discover other peers in the channel and facilitate communication between them. Only anchor peers can interact with other organizations' anchor peers. Peers can be part of one or more channels and have multiple IoMT (Internet of Medical Things) devices connected to them. These low-processing power devices generate data that is sent to the nearby connected data collector peer node, which then adds the data to the blockchain for further use. Lastly, the Trusted Authority (TA) is a trusted entity responsible for authenticating and registering all entities during the mutual authentication and key agreement phase.

3.3. ASCON

According to the Competition for Authenticated Encryption: Security, Applicability, and Robustness (CEASER), ASCON has been chosen as the "primary choice" for lightweight authenticated encryption [19]. ASCON is an online Authenticated Encryption with Associated Data (AEAD) scheme that ensures the data's authenticity, confidentiality, and integrity all at the same time. ASCON's encryption process can be described using the given expression: $(CT, Tag) = \epsilon_K\{AD, N, PT\}$, where PT, CT, N, K, AD, Tag implies, plain text, cipher text, nonce, key, associative data and authentication parameter (Tag), respectively. ASCON's decryption process can be described using the following expression: $(PT, Tag^*) = D_K\{AD, N, CT\}$, where N, AD, PT, CT, Tag^* implies nonce, associative data, plain text, cipher text and authentication parameter (Tag), respectively. The tag can also be used to verify the authenticity of the retrieved plaintext using, $Tag^* = Tag$. ASCON is employed as the encryption/decryption algorithm in the proposed BioMTAKE model.

4. Proposed scheme

This section presents the details of the proposed scheme.

4.1. Overview

Fig. 3 shows the complete overview of the proposed scheme.

4.1.1. Initialization of blockchain

Configuring network model in Hyperledger Fabric by updating cryptotgen.yaml, configtx.yaml, and orderer.yaml files. Next, Generate all the necessary keys for registration of the node and the orderer, which is stored with the MSP (Membership Service Provider) of the respective node. The MSP keeps track of all the keys generated for participating nodes and verifies the signature of the peer whenever a transaction request is submitted from the peer. Then, we set up the genesis block, which is the first block of the blockchain network. Next is the creation of a channel for transactions. Channel is where the connection of the peers is established, and the transaction occurs.

4.1.2. Adding peers to the network

Using the certificate and keys in the MSP of the node, the node requests the admin node for joining the network. Using the authentication method of the Hyperledger Fabric, the admin node authenticates the node. Once the node is set up and the network is up and running, the network is ready for accepting the transactions on the blockchain through the channel created during the configuration setup.

4.1.3. IoT/user authentication using key agreement scheme

Register the IoMT device to the node that it is going to interact with and the trusted authority. If the IoMT device wants to participate in the communication it has to begin a secure session. The IoMT device is authenticated using the proposed authentication method, which is lightweight and privacy-preserving. Using the key agreement method, the IoMT device is authenticated by the data collecting node that it wants to interact with. Once the IoMT device is authenticated by the data collector node, a session has been established, and a secure channel is created for further communications.

4.2. Detailed steps of the proposed scheme

The proposed scheme BioMTAKE comprises three phases, which are (A) Initial system setup, (B) Blockchain Initialization, (C) IoMT device Registration phase, and (D) Mutual authentication and key exchange. It is assumed that all the nodes and entities in the IoMT environment are time-synchronized. BioMTAKE implements an AEAD (Authenticated Encryption with Associated Data) scheme using ECC, hash function, and ASCON to design a robust Authentication protocol for the IoMT environment. Fig. 2 shows the notations used in the proposed scheme. A detailed description of all phases of BioMTAKE is presented as follows:

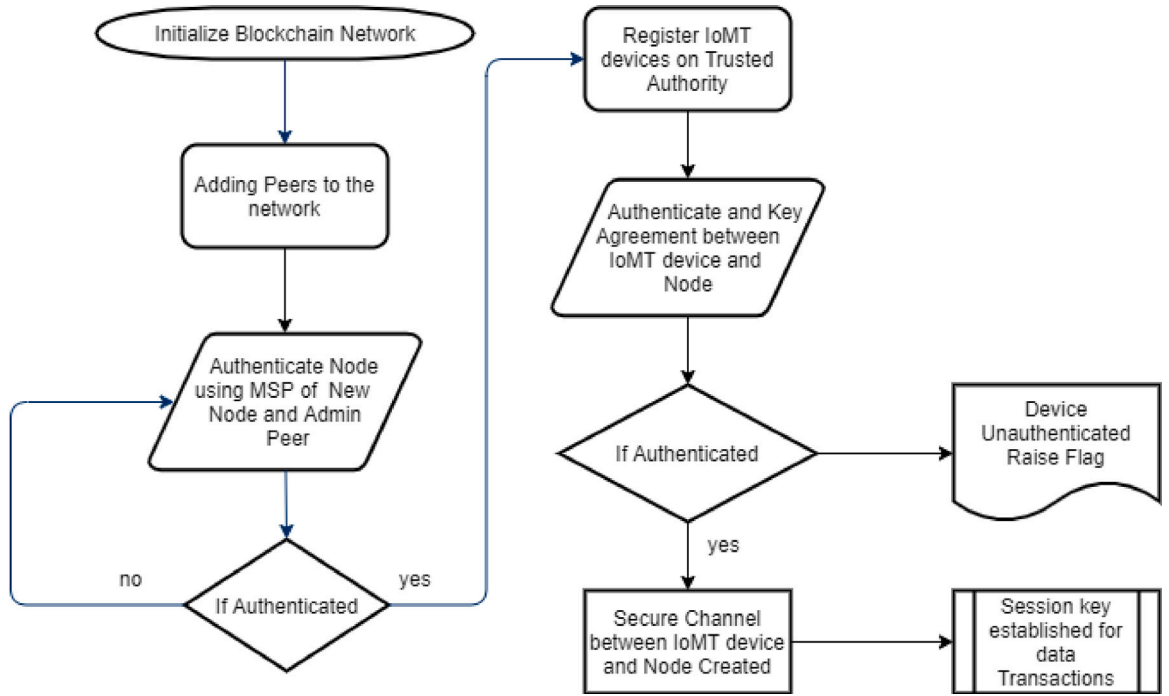


Fig. 3. Flow diagram of the process.

4.2.1. Initial system setup

TA selects an elliptic curve $E_p(a, b)$ over Z_q^* and $4a^3 + 27b^2 \neq 0 \pmod{p}$ where $a, b \in Z_q^*$ and P is generator and prime order q .

- Initially, TA randomly chooses $SK_{TA} \in Z_q^*$ as secret key and calculates $Pb_{TA} = SK_{TA} \cdot P$ as public Key.
- TA selects hash function h using the ASCON hash function.
- For a specific DC node with identity as ID_{DC} , computes secret Key $SK_{DC} \in Z_q^*$, $Pb_{DC} = SK_{DC} \cdot P$ as public Key.
- Lastly, TA publishes system public parameters as $parm = \{P, q, Pb_{TA}, h\}$ and keeps SK_{TA} secret.

4.2.2. Blockchain initialization phase

The Hyperledger Fabric network is initialized with the organization, orderers, peers, and Certificate Authorities (CAs). Each data collector node acts as a peer in its respective organization, with one set as an anchor peer. The administrator creates two channels, one for local data and one for global data, and adds the appropriate peers to each channel. The local channel has the localchain chaincode, which holds information about IoMT devices associated with a data collector node, while the global channel has the globalchain chaincode, which stores data accessible by multiple organizations. Transactions initiated by users through peers are written into a newly generated block of the blockchain once the network is built. To ensure the ledger's integrity, modifications to the ledger or chaincode must be approved according to the consensus of the participating peers. Peers use the functions provided by Hyperledger Fabric to interact with the blockchain, perform operations, and execute transactions by invoking chaincode.

4.2.3. IoMT device registration phase

Here, the device registration steps are shown.

- IoMT device retrieves its ID_{IM} from its processor. Also generates a random number $r1$. Both ID_{IM} and $r1$ are 128 bits each. It calculates public key $Pb_{IM} = r1 \cdot P$. Then IoMT sends registration request message $MSG1 = \{ID_{IM}, Pb_{IM}\}$ to TA.
- After receiving the message $MSG1$, TA calculates secret parameter $SP = h(SK_{DC} \parallel ID_{DC})$, pseudo identity for IoMT device $PID_{IM} = h(ID_{IM} \parallel SK_{TA} \cdot Pb_{IM})$, plain text $PT_{DC} = (Pb_{IM} \parallel ID_{IM})$, key for ASCON encryption $K = SP_a \oplus SP_b$, where SP_a and SP_b are two chunks of SP , each of 128 bits and nonce $N_{DC} = SP_a$. TA then computes $(CT_{DC}, Tag_{DC}) = \epsilon_K\{N_{DC}, PT_{DC}\}$ using the ASCON encryption. Where CT_{DC} and Tag_{DC} are the ciphertext and authentication parameters. TA store $\{PID_{IM}, CT_{DC}, Tag_{DC}, Pb_{DC}\}$ into the globalchain. TA returns $\{PID_{IM}\}$ to IoMT.
- IoMT adds $\{h(Pb_{IM}), PID_{IM}\}$ to the localchain.

Algorithm 1: Chaincode function for insert operation

```

func (s *SmartContract) createAsset(ctx, hpbm, pidm):
    avail, Err = s.AssetExists(ctx, hpbm)
    if Err!=nil then
        | return error
    end
    if avail then
        | return fmt.Errorf("IoMT device exists", hpbm)
    end
    asset = Asset{HPBIM: hpbm, PIDM: pidm}
    assetJSON, Err = json.Marshal(asset)
    if Err!=nil then
        | return error
    end
    return ctx.GetStub().PutState(hpbm, assetJSON)
End Function

```

Algorithm 2: Chaincode function for update operation

```

func (s *SmartContract) updateAsset(ctx, hpbm, pidm):
    avail, Err = s.AssetExists(ctx, hpbm)
    if Err!=nil then
        | return error
    end
    if !avail then
        | return fmt.Errorf("IoMT device does not exist", hpbm)
    end
    asset = Asset{HPBIM: hpbm, PIDM: pidm}
    assetJSON, Err = json.Marshal(asset)
    if Err!=nil then
        | return error
    end
    return ctx.GetStub().PutState(hpbm, assetJSON)
End Function

```

Algorithm 3: Chaincode function for delete operation

```

func (s *SmartContract) deleteAsset(ctx, hpbm):
    avail, Err = s.AssetExists(ctx, hpbm)
    if Err!=nil then
        | return error
    end
    if !avail then
        | return fmt.Errorf("IoMT device does not exist", hpbm)
    end
    return ctx.GetStub().DelState(hpbm)
End Function

```

4.2.4. Mutual authentication and key exchange phase

In this BioMTAKE phase, the IoMT device achieves the authentication with the data collector node and establishes a common secret session key SS with DC for secure communication in the future. To establish an SS , both IoMT and DC are required to execute the following steps (see Fig. 4):

1. IoMT device selects SK_{IM} , S_x , TM_x of 160, 128, 32 bits and computes $SK_{DC-IM} = SK_{IM} \cdot Pb_{DC}$ as the shared secret key of DC and IoMT. Moreover, it also computes $Pb_{IM1} = SK_{IM} \cdot P$ and makes it public. Further, computes $\alpha = h(PID_{IM} \parallel SK_{DC-IM} \parallel TM_x \parallel h(Pb_{IM}))$. IoMT splits α into α_a , α_b each with 128 bits. Then computes $K_{IM} = \alpha_a \oplus \alpha_b$ and $N_x = \alpha_a$. Furthermore, using ASCON encryption algorithm IoMT computes $(CT_x, Tag_x) = e_{K_{IM}}\{N_x, S_x\}$. Where CT_x and Tag_x are the ciphertext and authentication parameters. Finally, IoMT sends $\{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$ to DC .
2. Data collector node first compares the timestamp $\Delta T \geq |TM^* - TM_x|$. DC terminates the process if the above condition fails. Otherwise, DC computes $SK_{DC-IM} = SK_{DC} \cdot Pb_{IM1}$ as shared secret key of DC and IoMT. DC extracts PID_{IM} using $h(Pb_{IM})$ from the ledger. Next, it computes, $\beta = h(PID_{IM} \parallel SK_{DC-IM} \parallel TM_x \parallel h(Pb_{IM}))$. DC splits β into β_a , β_b each

Algorithm 4: Chaincode function for read operation

```

func (s *SmartContract) readAsset(ctx, hpbm):
    assetJSON, Err := ctx.GetStub().GetState(hpbm)
    if Err!=nil then
        | return fmt.Errorf("Failed")
    end
    if assetJSON == nil then
        | return fmt.Errorf("IoMT device does not exist", hpbm)
    end
    var asset Asset
    return &asset, nil
End Function

```

with 128 bits and computes $K_{DC} = \beta_a \oplus \beta_b$ and $N_y = \beta_a$. Furthermore, using ASCON encryption algorithm IoMT computes $(PT_y, Tag_y) = D_{K_{DC}}\{N_y, CT_x\}$. Finally, $Tag_y = Tag_x$ is validated by DC to check the authenticity of the message received. If the condition holds true, DC says messages as valid and extracts $PT_y = S_x$ from decryption. Otherwise, DC terminates the authentication process.

- After validation message authenticity, DC computes $SP_z = h(SK_{DC} \parallel ID_{DC})$ and splits SP_z into SP_{za}, SP_{zb} each with 128 bits. Retrieves $\{CT_{DC}, Tag_{DC}\}$ from the globalchain ledger using the PID_{IM} . Further, computes $K_1 = SP_{za} \oplus SP_{zb}$, $N_{DC} = SP_{za}$, $((Pb_{IM} \parallel ID_{IM}), Tag_{DC}^*) = D_{K_1}\{N_{DC}, CT_{DC}\}$. To validate the authenticity of the data from the ledger, DC checks the condition $Tag_{DC}^* = Tag_{DC}$. If the condition holds true, DC continues the process. Otherwise, DC terminates the authentication process. DC selects TM_z, S_z of size 32 and 128 bits and calculates $\gamma = h(SK_{DC-IM} \parallel ID_{IM} \parallel h(Pb_{IM}) \parallel TM_z \parallel S_x)$ and splits γ into γ_a, γ_b each with 128 bits. $AD_z = \gamma_a \oplus \gamma_b$, $N_z = \gamma_a$, $PT_z = (PID_{IM} \parallel S_z)$, where AD_z, N_z, PT_z associative data, nonce, and plaintext, respectively. Then, using ASCON encryption algorithm, DC calculates $(CT_z, Tag_z) = e_{Pb_{IM}}\{AD_z, N_z, PT_z\}$. Also, DC calculates the secret session key by computing $SS_{DC} = h(\gamma \parallel S_x \parallel S_z \parallel TM_z)$ for secure communication in the future. Finally, DC composes a message $\{TM_z, CT_z, Tag_z\}$ and sends it to IoMT via the open channel.
- After receiving the message, the IoMT device compares the timestamp $\Delta T \geq |TM^* - TM_z|$. IoMT terminates the process if the above condition fails. Otherwise, IoMT computes $\delta = h(SK_{DC-IM} \parallel ID_{IM} \parallel h(Pb_{IM}) \parallel TM_z \parallel S_x)$ and splits δ into δ_a, δ_b each with 128 bits. $AD_p = \delta_a \oplus \delta_b$, $N_p = \delta_a$, where AD_p, N_p are associative data and nonce respectively. Then, using ASCON decryption algorithm, IoMT calculates $(PT_z, Tag_p) = D_{Pb_{IM}}\{AD_p, N_p, CT_z\}$. IoMT checks the condition $Tag_p = Tag_z$. If the condition holds true, IoMT continues the process. Otherwise, terminates the authentication process. IoMT then retrieves S_z from PT_z by $PT_z = (PID_{IM} \parallel S_z)$. Finally, IoMT calculates the secret session key by computing $SS_{IM} = h(\delta \parallel S_x \parallel S_z \parallel TM_z)$ for secure communication in the future.

This ends the authentication process and thus, IoMT and DC establish a shared secret session key. In future communications, both IoMT devices and DC use the common session key for secure data transfer.

5. Security analysis

This section outlines the security analysis of the proposed BioMTAKE scheme. Formal Analysis is performed by using the ROR model that ensures the semantic security of the Session Key. The Scyther tool is then used to demonstrate that BioMTAKE is protected against different covert threats. An informal analysis is provided in order to demonstrate the immunity towards various security attacks. Initially, an informal assessment is provided to demonstrate that BioMTAKE is immune to several security threats, such as impersonation, MITM, and replay.

5.1. Formal security analysis

Formal Security Analysis is done using the RoR Model that ensures the semantic security of the session key generated between IoMT devices and the Data Collector Node. This analysis is performed to verify the mutual authentication of the BioMTAKE protocol. There are two participants involved—IoMT devices (π_i^M) and Data Collector Node (π_j^{DC}). Various queries used for the real attack by the adversary A in this analysis are-

- $EavesDrop(\pi_i^M, \pi_j^{DC})$: By using this query, adversary A eavesdrops on the communication between the IoMT devices and the Data Collector node. When a challenger C receives this query, it returns all the messages exchanged by π_i^M and π_j^{DC} .
- $Send(\pi_i^X, m)$: This is a query where A sends message m to challenger C and gets back the reply according to the actual protocol.
- $CorruptIoMT(\pi_i^X)$: This is used by the A in order to corrupt the given instance and get all the information stored there i.e., long term secrets.
- $Reveal(\pi_i^X)$: By this query the adversary A get the session key in response if the given instance π_i^X is in the accepted state.
- $Toss(\pi_i^X)$: This query is performed only once and proves the semantic security of the session key. This query is like tossing a coin 'c' that gives two values. It results in a random number if $c = 0$ and a session key if $c = 1$.

IoMT Device (IM)	Data Collector Node (DC)
<p>Step 1: select SK_{IM}, S_x, TM_x of 160, 128, 32 bits</p> <p>$Pb_{IM} = SK_{IM} \cdot P$</p> <p>$SK_{DC-IM} = SK_{IM} \cdot Pb_{DC}$</p> <p>$\alpha = h(PID_{IM} SK_{DC-IM} TM_x h(Pb_{IM}))$</p> <p>splits α into α_a, α_b</p> <p>$K_{IM} = \alpha_a \oplus \alpha_b$, $N_x = \alpha_a$</p> <p>$(CT_x, Tag_x) = \mathcal{E}_K \{N_x, S_x\}$</p> <p>sends $M1 = \{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$</p>	
<p>Step 2: $\Delta T \geq TM^* - TM_x$ if false terminate.</p> <p>$SK_{DC-IM} = SK_{DC} \cdot Pb_{IM}$</p> <p>Extracts PID_{IM} using $h(Pb_{IM})$ from the ledger</p> <p>$\beta = h(PID_{IM} SK_{DC-IM} TM_x h(Pb_{IM}))$</p> <p>splits β into β_a, β_b</p> <p>$K_{DC} = \beta_a \oplus \beta_b$ and $N_y = \beta_a$</p> <p>$(PT_y, Tag_y) = \mathcal{D}_{K_{DC}} \{N_y, CT_x\}$</p> <p>Check if $Tag_y = Tag_x$, if false terminate process.</p> <p>Extract $\{S_x\} = PT_y$</p>	
<p>Step 3: $SP_z = h(SK_{DC} IDDC)$ splits SP_z into SP_{za}, SP_{zb}</p> <p>Retrieves $\{CT_{DC}, Tag_{DC}\}$ from the globalchain ledger</p> <p>$K_1 = SP_{za} \oplus SP_{zb}$, $N_{DC} = SP_{za}$</p> <p>$((Pb_{IM} ID_{IM}), Tag_{DC}^*) = \mathcal{D}_{K_1} \{N_{DC}, CT_{DC}\}$</p> <p>Validate $Tag_{DC}^* = Tag_{DC}$, if false terminate.</p> <p>DC selects TM_z, S_z of size 32 and 128 bits</p> <p>$\gamma = h(SK_{DC-IM} ID_{IM} h(Pb_{IM}) TM_z S_x)$</p> <p>splits γ into γ_a, γ_b each with 128 bits.</p> <p>$AD_z = \gamma_a \oplus \gamma_b$, $N_z = \gamma_a$, $PT_z = (PID_{IM} S_z)$</p> <p>$(CT_z, Tag_z) = \mathcal{E}_{Pb_{IM}} \{AD_z, N_z, PT_z\}$</p> <p>$SS_{DC} = h(\gamma S_x S_z TM_z)$</p> <p>send $M2 = \{TM_z, CT_z, Tag_z\}$</p>	
<p>Step 3: $\Delta T \geq TM^* - TM_z$ if false terminate.</p> <p>$\delta = h(SK_{DC-IM} ID_{IM} h(Pb_{IM}) TM_z S_x)$</p> <p>splits δ into δ_a, δ_b each with 128 bits</p> <p>$AD_p = \delta_a \oplus \delta_b$, $N_p = \delta_a$</p> <p>$(PT_z, Tag_p) = \mathcal{D}_{Pb_{IM}} \{AD_p, N_p, CT_z\}$</p> <p>Check $Tag_p = Tag_z$, if false terminate process.</p> <p>$\{PID_{IM}, S_z\} = PT_z$</p> <p>$SS_{IM} = h(\delta S_x S_z TM_z)$</p>	

Fig. 4. Mutual authentication and key exchange.

Definition 1 (Accepted State). If the instance π_i^X reaches the final accepted state after receiving the last message of actual protocol, it is in the accepted state. A session identifier is constructed by rearranging all the messages in a sequence.

Definition 2 (Partnership). Two instances π_i^X and $\pi_i^{\bar{X}}$ are partners if they (1) both are in the accepted state, (2) both mutually authenticate each other and compute a session key, (3) both are mutual partners.

Definition 3 (Freshness). Two instances π_i^X and $\pi_i^{\bar{X}}$ are fresh if their established common session key is not revealed to any adversary A.

The proposed protocol BioMTAKE can be attacked by the adversary A only when A guesses and authenticate the value of Tag_y, Tag_{DC}, Tag_p correctly. So, an advantage in guessing the security of the session key is given by $Adv_A^{BioMTAKE}(t)$. Probability of breaking the proposed protocol BioMTAKE is given by $Adv_A^{CDH}(t)$.

Theorem 5.1. The session key security of the proposed protocol can be broken by the adversary A in time t when the advantage $Adv_A^{BioMTAKE}(t)$ of A is given by-

$$Adv_A^{BioMTAKE}(t) \leq \frac{q_{num_hash}^2}{|Hash|} + \frac{2 \cdot q_{num_send}}{|ID|} + 2 \cdot Adv_A^{CDH}(t)$$

where q_{num_hash} is the number of query by A for hash function, q_{num_send} is the number of query by A for sending message, $|Hash|$ is the range value of the hash function, $|ID|$ is the size of the identity dictionary. $Adv_A^{CDH}(t)$ shows the advantage of adversary A for breaking the CDH problem in time t .

Proof. This can be proved by taking the four game scenario given by Game 0, Game 1, Game 2, Game 3, and Game 4. P_j denotes the winning of the game by the adversary by guessing the result of $Toss(\pi_i^X)$. $Pr[P_j]$ denotes the probability of winning the game by the adversary. The game scenario is -

- **Game 0:** In Game 0, the adversary begins by the real attack on the proposed protocol and tries to guess the value of Tag_y, Tag_{DC}, Tag_p . No queries are involved while guessing these parameters. So,

$$Adv_A^{BioMTAKE}(t) = |2 \cdot Pr[P_0] - 1| \quad (1)$$

- **Game 1:** Everything is similar to Game 0 except that the adversary A tries to eavesdrop and use query $EavesDrop(\pi_i^M, \pi_j^{DC})$. This help A to use message communicated between the involved participants like $\{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$, $\{TM_z, CT_z, Tag_z\}$. Then, A uses $Toss(\pi_i^X)$ query to obtain its value. It results in a common session key if the value is 1 and a random number if its value is 0. Session key calculated by Data Collector Node is given by $SS_{DC} = h(\gamma \| S_x \| S_z \| TM_z)$ and session key calculated by IoMT devices is given by $SS_{IM} = h(\delta \| S_x \| S_z \| TM_z)$. To find session key, A requires γ, δ or random number S_x, S_z . So, there is no chance of winning Game 1.

$$Pr[P_0] = Pr[P_1] \quad (2)$$

- **Game 2:** In this scenario, A uses $hash()$ and $Send(\pi_i^X, m)$ query. Then, A uses the message communicated between two participants $\{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$, $\{TM_z, CT_z, Tag_z\}$ for getting information of the secret session key. Session key contains γ, δ, S_x, S_z . The value of γ, δ is protected by the hash function. A needs to find out these values which is not possible as randomness ensures no collision in the $hash()$ value. So,

$$|Pr[P_1] - Pr[P_2]| \leq \frac{q_{num_hash}^2}{2 \cdot |Hash|} \quad (3)$$

- **Game 3:** In this game scenario, A uses $CorruptIoMT(\pi_i^X)$ query and tries to guess the real identity ID_{IM} of the IoMT devices. In order to find out the session key, A needs γ or δ that requires the value of ID_{IM} . For guessing ID_{IM} , A guesses the value from identity dictionary $|ID|$. So,

$$|Pr[P_2] - Pr[P_3]| \leq \frac{q_{num_send}}{|ID|} \quad (4)$$

- **Game 4:** A tries to get secret session key SS_{DC}, SS_{IM} by $EavesDrop(\pi_i^M, \pi_j^{DC})$ and get information like $\{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$, $\{TM_z, CT_z, Tag_z\}$. However, A cannot get information of γ or δ as it involves hash digest and SK_{DC-IM} , which contains the random number SK_{IM} . To compute SK_{DC-IM} , the adversary need to solve CDH problem in time t . Also, A cannot guess the random number S_x, S_z correctly. Therefore,

$$|Pr[P_3] - Pr[P_4]| \leq Adv_A^{CDH}(t) \quad (5)$$

After execution of all the game, A tries to guess the result of $Toss(\pi_i^X)$ which has two outcomes 0 or 1. So,

$$Pr[P_4] = \frac{1}{2} \quad (6)$$

Dividing Eq. (1) by 2,

$$\frac{Adv_A^{BloMTAKE}(t)}{2} = |Pr[P_0] - \frac{1}{2}|$$

From equation (2),

$$\frac{Adv_A^{BloMTAKE}(t)}{2} = |Pr[P_1] - \frac{1}{2}| \quad (7)$$

From equation (6),

$$\frac{Adv_A^{BloMTAKE}(t)}{2} = |Pr[P_1] - Pr[P_4]|$$

Applying Difference Lemma,

$$\begin{aligned} |Pr[P_1] - Pr[P_4]| &\leq |Pr[P_1] - Pr[P_3]| + |Pr[P_3] - Pr[P_4]| \\ &\leq |Pr[P_1] - Pr[P_2]| + |Pr[P_2] - Pr[P_3]| \\ &\quad + |Pr[P_3] - Pr[P_4]| \end{aligned}$$

From equation (3), (4), (5) :

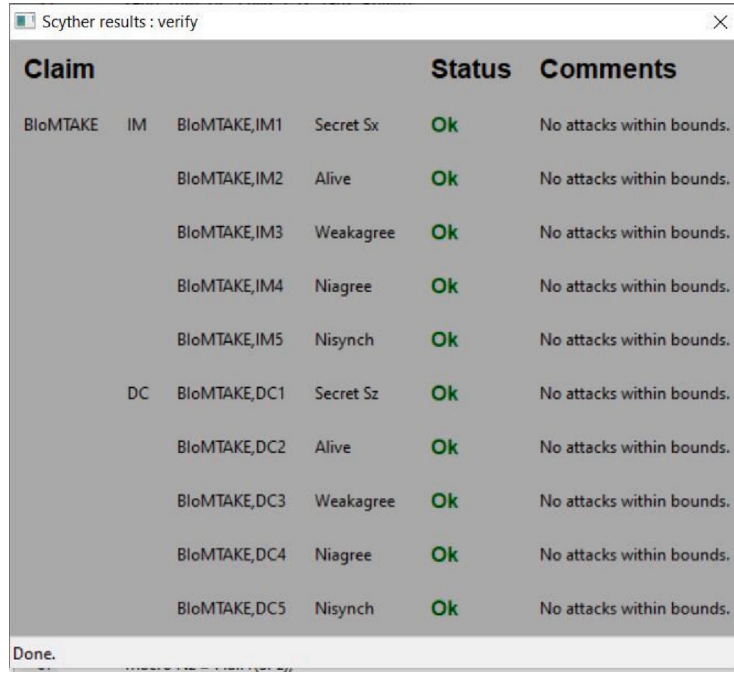
$$\leq \frac{q_{num_hash}^2}{2 \cdot |Hash|} + \frac{q_{num_send}}{|ID|} + Adv_A^{CDH}(t)$$

From Eqs. (7), (8),

$$\begin{aligned} \frac{Adv_A^{BloMTAKE}(t)}{2} &\leq \frac{q_{num_hash}^2}{2 \cdot |Hash|} + \frac{q_{num_send}}{|ID|} + Adv_A^{CDH}(t). \\ Adv_A^{BloMTAKE}(t) &\leq \frac{q_{num_hash}^2}{|Hash|} + \frac{2 \cdot q_{num_send}}{|ID|} + 2 \cdot Adv_A^{CDH}(t). \quad \square \end{aligned}$$

5.2. Informal security analysis

- Anonymity and Untraceability:** There are two messages $MSG1 = \{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$ and $MSG2 = \{TM_z, CT_z, Tag_z\}$ which are exchanged during the authentication phase in BloMTAKE scheme. Pseudo identity of the IoMT device PID_{IM} is calculated by $PID_{IM} = h(ID_{IM} \parallel SK_{TA} \cdot Pb_{IM})$ where ID_{IM} is the real identity of the IoMT device and SK_{TA} is a secret key of TA and Pb_{IM} is public Key of IoMT device. Pb_{IM} is generated fresh on every new session by a random SK_{IM} value secret Key of IoMT device. Furthermore, $\alpha = h(PID_{IM} \parallel SK_{DC-IM} \parallel TM_x \parallel h(Pb_{IM}))$ so α cannot extract PID_{IM} . On every session, a new PID_{IM} is generated, also it is not possible to extract PID_{IM} from any operation in the later phase. Therefore, it suggests that BloMTAKE provides an anonymity feature. Moreover, MSG1 and MSG2 in BloMTAKE vary dynamically and unpredictably for each new authentication session, making it extremely difficult for an attacker to associate recorded messages, such as MSG1 and MSG2, from two distinct authentication sessions in order to extract any relevant information.
- Session Key Agreement:** All participants compute share secret key SS as $SS = h(\delta \parallel S_x \parallel S_z \parallel TM_z)$, where δ , S_x , S_z and TM_z are calculated during each session. As a result, the proposed scheme establishes a shared session key.
- Offline TA:** TA is not involved in the proposed approach during the authentication and key exchange phases. The data remains on the blockchain, and authentication and revocation are accomplished by triggering blockchain transactions or the particular device or node.
- Impersonation attack:** To complete the authentication phase in BloMTAKE, two messages, such as $MSG1 = \{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$ and $MSG2 = \{TM_z, CT_z, Tag_z\}$, are sent. By producing a valid MSG1, the attacker may spoof a real IoMT device. Although, without acquiring the secret parameters, such as SK_{IM} , SK_{DC-IM} , PID_{IM} , and hash function, the Attacker cannot produce an acceptable message MSG1 on behalf of devices. As a result, the attacker cannot carry out the device impersonation attack. Likewise, without acquiring the secret credentials of DC, such as SK_{DC} , SK_{DC-IM} , ID_{IM} , ID_{DC} , the attacker cannot impersonate DC. As a result, the suggested BloMTAKE can withstand an IM/DC impersonation attack.
- Man-in-the-middle Attack:** The attacker can intercept all messages exchanged throughout the authentication process, including $MSG1 = \{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$ and $MSG2 = \{TM_z, CT_z, Tag_z\}$. After intercepting MSG1, the attacker can produce a fake message MSG1' to trick DC into assuming that MSG1' is from an authentic IM. Although, without acquiring the secret credentials SK_{IM} , SK_{DC-IM} , and PID_{IM} , the attacker will be unable to create a valid MSG1. Likewise, for DC without acquiring the secret credentials, such as SK_{DC} , SK_{DC-IM} , ID_{IM} , and ID_{DC} , it is challenging for Attacker to produce MSG2. This shows that BloMTAKE scheme can withstand MITM attacks.
- Replay Attack:** The attacker can intercept all messages exchanged throughout the authentication process, including $MSG1 = \{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$ and $MSG2 = \{TM_z, CT_z, Tag_z\}$. After intercepting, the attacker might try to replay the recorded communication to network entities in order to get sensitive details from the entities. In BloMTAKE, all network elements that are engaged in the authentication step are time synchronized. Furthermore, all transmitted messages, namely $MSG1$ and $MSG2$, include the most recent timestamp and randomly generated number. An entity that is receiving message checks



Claim				Status	Comments
BloMTAKE	IM	BloMTAKE,IM1	Secret Sx	Ok	No attacks within bounds.
		BloMTAKE,IM2	Alive	Ok	No attacks within bounds.
		BloMTAKE,IM3	Weakagree	Ok	No attacks within bounds.
		BloMTAKE,IM4	Niagree	Ok	No attacks within bounds.
		BloMTAKE,IM5	Nisynch	Ok	No attacks within bounds.
DC		BloMTAKE,DC1	Secret Sz	Ok	No attacks within bounds.
		BloMTAKE,DC2	Alive	Ok	No attacks within bounds.
		BloMTAKE,DC3	Weakagree	Ok	No attacks within bounds.
		BloMTAKE,DC4	Niagree	Ok	No attacks within bounds.
		BloMTAKE,DC5	Nisynch	Ok	No attacks within bounds.

Done.

Fig. 5. Scyther security analysis of BioMTAKE.

the condition $\Delta T \geq |TM^* - TM_x|$ and $\Delta T \geq |TM^* - TM_z|$ for $MSG1$ and $MSG2$, respectively, to validate the freshness of received message. In the event of a failure, the relevant entity discards the received message and ends the authentication procedure. The entity that receives the message can identify the replay assault in this manner. As a result, BioMTAKE can withstand a replay attack.

7. **IoMT Device Capture Attack:** By using the power analysis attack, an attacker can gain access to an IoMT device installed in the network system and retrieve secret parameters like ID_{IM} , hash algorithm, random number generator, that are stored in the memory of the IoMT device. Although using the secret credentials retrieved from the captured IoMT device, the attacker cannot access the secret credentials of other non-compromised or non-captured IoMT devices and Data collector nodes. Furthermore, the secret credentials for each deployed IoMT device are distinctive. As a result, the attacker cannot endanger the safety of the IoMT device that is not captured. As a result, the suggested BioMTAKE scheme can withstand IoMT device capture attacks.
8. **Distributed Database Access and Storage:** Distributed database access and storage are achieved by setting up the Hyperledger fabric blockchain network on data collector nodes across several organizations and interpreting it as fabric peers. These Peers engage in the consensus protocol, and no modifications may indeed be made without peer approval according to the endorsement policy. As a result, the suggested BioMTAKE scheme allows for distributed database access and storage.

5.3. Scyther security analysis

A tool for formal verification Scyther [20] is being used to investigate the design flaws and peculiarities of BioMTAKE. Scyther implements the proposed security protocol using security protocol description language (SPDL), similar to Python language, and the semantics specified in [20]. Scyther attempts to achieve the following for an SPDL-defined security protocol: (i) confirmation of the claims described in the specified protocol, (ii) confirmation of security claims created automatically by the Scyther tool, and (iii) thorough categorization of the stated roles. In BioMTAKE, there are two roles described: IM and DC. Fig. 5 clearly shows that BioMTAKE is secure. Fig. 5 clearly indicates that the presented claims in the SPDL script, that is $Claim(IM, Secret, S_x)$, $Claim(DC, Secret, S_y)$, and other claims, such as weak agreement, aliveness, Non-injective synchronization (Nisynch), and Non-injective agreement (Niagree), are all “OK” and free from all possible attacks.

6. Performance analysis

In this section, we investigate the proposed BioMTAKE scheme’s overall performance in relation to communication and computational overhead. In addition, we compare the suggested scheme to schemes represented in the papers of Bera et al. [23], Das et al. [24] and Bera et al. [25] in order to demonstrate its computational and communication efficiency.

Table 1

Cryptographic operations's computational time comparison.

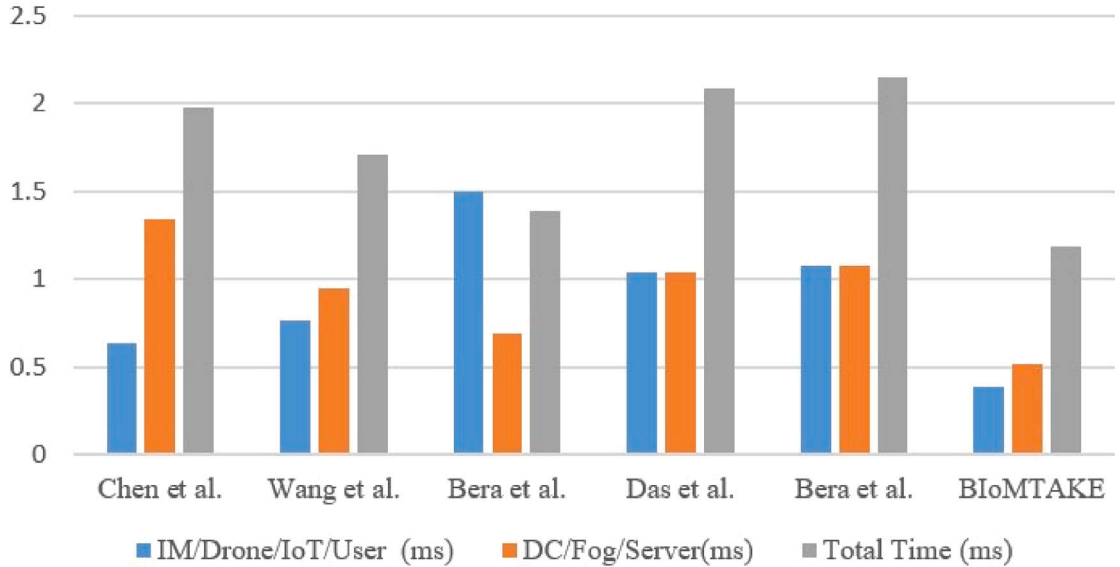
Protocol/Scheme	IM/Drone/IoT/User	DC/Fog Server	Total Time
Chen et al. [21]	$10T_H \approx 0.637$ ms	$15T_H + 6T_H \approx 1.3377$ ms	$31T_H \approx 1.9747$ ms
Wang et al. [22]	$6T_H + 4T_{ECC} + 3T_{ECA} \approx 0.7634$ ms	$6T_H + 6T_{ECC} + 3T_{ECA} \approx 0.9492$ ms	$12T_H + 10T_{ECC} + 6T_{ECA} \approx 1.7126$ ms
Bera et al. [23]	$5T_H + 4T_{ECC} + T_{ECA} + T_{PO} \approx 1.4933$ ms	$5T_H + 4T_{ECC} + T_{ECA} \approx 0.6933$ ms	$10T_H + 8T_{ECC} + 2T_{ECA} \approx 1.386$ ms
Das et al. [24]	$6T_H + 7T_{ECC} + 3T_{ECA} \approx 1.0421$ ms	$6T_H + 7T_{ECC} + 3T_{ECA} \approx 1.0421$ ms	$12T_H + 14T_{ECC} + 6T_{ECA} \approx 2.084$ ms
Bera et al. [25]	$11T_H + 4T_{ECC} + T_{ECA} \approx 1.0755$ ms	$11T_H + 4T_{ECC} + T_{ECA} \approx 1.0755$ ms	$22T_H + 8T_{ECC} + 2T_{ECA} \approx 2.151$ ms
BloMTAKE	$4T_H + 2T_{ECC} + T_{AE} + T_{AD} \approx 0.3874$ ms	$4T_H + T_{ECC} + T_{AE} + 2T_{AD} \approx 0.5177$ ms	$8T_H + 3T_{ECC} + 2T_{AE} + 3T_{AD} \approx 1.181$ ms

Table 2

Computational time of different cryptographic operations (in milliseconds).

Cryptographic Primitive	Notations	Time
ECC Point Multiplication	T_{ECC}	0.0929
ECC Point Addition	T_{ECA}	0.0032
ASCON Encryption	T_{AE}	0.0824
ASCON Decryption	T_{AD}	0.0438
ASCON Hash Function	T_H	0.0637

Computational Time Comparison

**Fig. 6.** Comparative analysis of the computational time.

6.1. Implementation setup

We have performed simulations on a laptop with an Intel i5-6200U CPU @ 2.30 GHz, 8 GB RAM running Ubuntu OS 18.04 LTS. Moreover, to determine ASCON execution time and different cryptographic operations, we utilize Cryptography and ASCON, a cryptography library based on Python. The running time of cryptographic operations is estimated by taking the mean of 10 successive executions of every operation with randomly generated inputs. We employ a non-singular elliptic curve $E_p(a, b)$ over Z_q^* and $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$ to evaluate ECC operations, where $a, b \in Z_q^*$ and P is a generator and prime order. Table 2 represents various cryptographic operations' computational time.

For blockchain implementations, we employed Hyperledger fabric v2.1 with Docker. To achieve consensus in our blockchain model, we have used the Raft consensus algorithm. A typical fabric network has three primary components that operate within docker containers during execution: orderer, peer node, and CA. In our implementation, we have deployed a fabric network with two organizations and one peer in each organization. Peers are connected through a channel. The chaincode is deployed on the channel between peers. The chaincode contains functions such createAsset, deleteAsset, updateAsset, and readAsset to perform insert, delete, update, and read operations, respectively. To communicate with the blockchain, perform operations, and execute

Table 3
Chaincode transactions performance evaluation.

Name	Max Latency (s)	Min Latency (s)	Avg Latency(s)	Throughput (TPS)
<i>createAsset</i>	7.37	0.87	5.27	143.6
<i>readAsset</i>	0.18	0.01	0.04	487.6
<i>updateAsset</i>	2.32	0.15	0.58	45.6
<i>deleteAsset</i>	0.19	0.01	0.04	437.6

Name	CPU% (max)	CPU% (avg)	Memory(max) [MB]	Memory(avg) [MB]	Traffic In [MB]	Traffic Out [MB]	Disc Write [MB]	Disc Read [B]
peer0.org2.example.com	65.14	45.20	118	106	8.01	4.46	6.39	0.00
peer0.org1.example.com	62.48	44.98	119	109	8.05	4.96	6.39	0.00
orderer.example.com	37.32	18.60	76.3	67.6	5.07	9.80	10.4	0.00

Fig. 7. Network peers resource utilization.

Table 4
Communication overhead comparison.

Schemes	IM/Drone/IoT/User to DC/Fog server	DC/Fog server to IM/Drone/IoT/User	Total
Chen et al. [21]	544 bits	2144 bits	2688 bits
Wang et al. [22]	928 bits	544 bits	1472 bits
Bera et al. [23]	864 bits	832 bits	1969 bits
Das et al. [24]	1664 bits	1632 bits	3296 bits
Bera et al. [25]	1472 bits	1568 bits	3040 bits
BloMTAKE	544 bits	288 bits	832 bits

transactions, peers must invoke chaincode using Hyperledger Fabric functions. The installed Hyperledger Fabric network model's performance is evaluated using Hyperledger Caliper, which initiates transactions via chaincode function calls.

6.2. Computational cost analysis

The computational cost of BloMTAKE and the corresponding authentication scheme is evaluated with the use of the computational time of various cryptographic operations, as shown in Table 1. Moreover, Fig. 6 shows the comparison between the existing scheme and the proposed scheme. BloMTAKE requires four hash operations, two ECC point multiplication operations, one ASCON encryption operation, and one ASCON decryption operation i.e., $4T_H + 2T_{ECC} + T_{AE} + T_{AD} \approx 0.3874$ ms at IOMT device. At DC BloMTAKE takes four hash operations, one ECC point multiplication operation, one ASCON encryption, and two ASCON decryption operations i.e., $4T_H + T_{ECC} + T_{AE} + 2T_{AD} \approx 0.5177$ ms, and overall BloMTAKE takes $8T_H + 3T_{ECC} + 2T_{AE} + 3T_{AD} \approx 1.181$ ms. Likewise, Table 1 and Fig. 6 show that BloMTAKE has less computational time than that of the comparable state-of-the-art schemes.

On the local system, we deploy the blockchain network model using the Hyperledger fabric instance. The execution times for various transactions are measured by running continuous multiple operations using the Hyperledger caliper performance tool. We run createAsset transaction 1000 times which adds 1000 assets to the ledger. Other transactions run for 60 s. The performance of the transactions is measured in terms of transaction latency and throughput. Here, transaction latency is the total time from transaction submission to the commitment of that transaction on the ledger and throughput is the total number of transactions committed to the ledger per unit of time. Table 3 and Fig. 7 shows the performance of Hyperledger fabric activities and peer resource utilization in the Hyperledger Fabric Network.

6.3. Communication cost analysis

The communication overhead of the proposed BloMTAKE system is described in this subsection. We analyze the number of messages exchanged and data sent through the communication channel between IM and DC to determine the communication overhead. Attachments delivered along with the initial message, such as ECC point, ID_{IM} , Tag, hash function, timestamp, and random number, add up to the communication cost. Two messages are exchanged during the authentication phase. Message $MSG1 : \{TM_x, CT_x, Tag_x, h(Pb_{IM})\}$ sent by IoMT device consumes $32 + 128 + 128 + 256 = 544$ bits and $MSG2 : TM_z, CT_z, Tag_z$ sent by DC consumes $32 + 128 + 128 = 288$ bits. This makes the total communication cost of the BloMTAKE scheme $544 + 288 = 832$ bits. In Table 4 and Fig. 8, we show the communication cost comparison between our scheme and the existing schemes. From Fig. 8 and Table 4, it is evident that BloMTAKE has less communication cost than that of the comparable state-of-the-art Schemes.

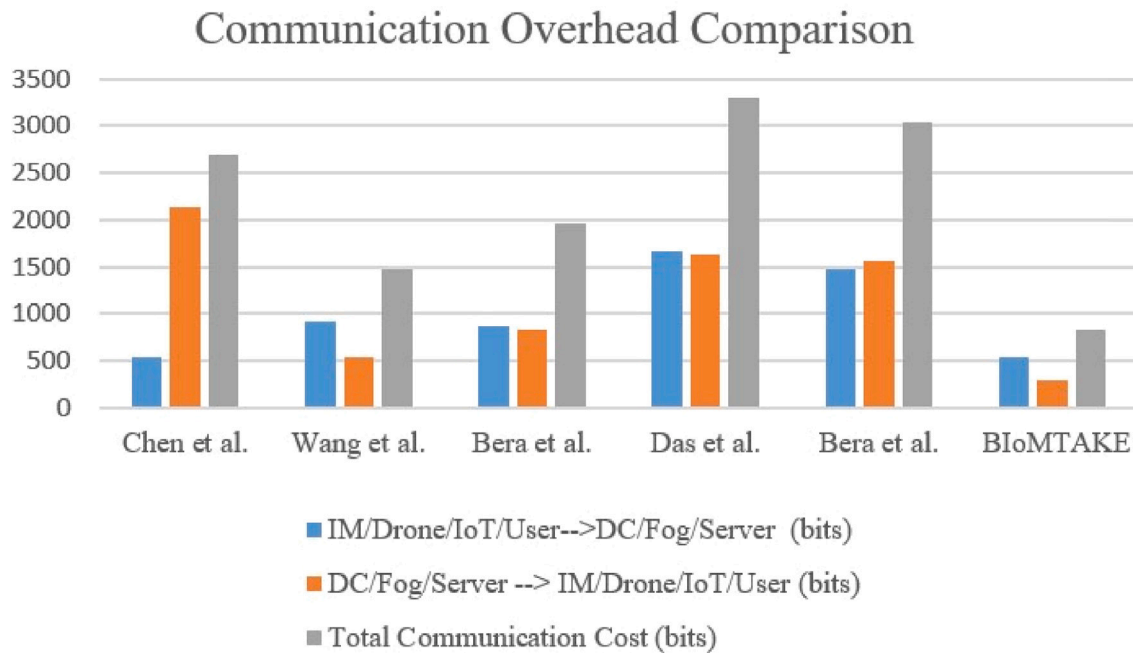


Fig. 8. Comparative analysis of the communication overhead.

7. Conclusion

The ever-growing user base and reliance on a single trusted authority expose the healthcare system to a variety of performance, reliability, security, and scalability challenges, including greater latency, single points of failure, centralization, and increased computing burden on the trusted authority. The advantages of blockchain have captivated both academics and organizations since its inception. In this paper, we investigate the use of Hyperledger Fabric blockchain as a distributed ledger technology to create a distributed healthcare environment that reduces the dependency on single trusted authority. We also present the BIoMTAKE scheme, a lightweight and privacy-preserving mutual authentication with a key agreement approach, to offer secured communication between IoMT devices and nodes (data collecting nodes). Our proposed scheme is designed to be safe against different stealthy attacks with lower computational and communication costs than that of equivalent existing schemes. We perform informal security analysis and Scyther-based analysis to establish the safety of our proposed scheme. Furthermore, our performance analysis based on Hyperledger Fabric and Cryptography library shows that the proposed BIoMTAKE scheme outperforms existing schemes. In the future, we plan to evaluate the authentication process of the proposed BIoMTAKE scheme on Raspberry Pi to see if it is feasible for real-time IoMT devices.

CRediT authorship contribution statement

Ashish Tomar: Data curation, Writing – original draft. **Niraj Gupta:** Data curation, Writing – original draft. **Divya Rani:** Data curation, Writing – original draft. **Sachin Tripathi:** Data curation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

- [1] P.K. Sadhu, V.P. Yanambaka, A. Abdelgawad, Internet of things: Security and solutions survey, *Sensors* 22 (19) (2022) 7433.
- [2] K.P. Satamraju, B. Malarkodi, A decentralized framework for device authentication and data security in the next generation internet of medical things, *Comput. Commun.* 180 (2021) 146–160.
- [3] B. Deebak, F.H. Memon, S.A. Khawaja, K. Dev, W. Wang, N.M.F. Qureshi, In the digital age of 5G networks: Seamless privacy-preserving authentication for cognitive-inspired internet of medical things, *IEEE Trans. Ind. Inform.* 18 (12) (2022) 8916–8923.
- [4] H.C. Van Tilborg, S. Jajodia, *Encyclopedia of Cryptography and Security*, Springer Science & Business Media, 2014.
- [5] N. Alsaeed, F. Nadeem, Authentication in the internet of medical things: Taxonomy, review, and open issues, *Appl. Sci.* 12 (15) (2022) 7487.
- [6] A. Tomar, S. Tripathi, Blockchain-assisted authentication and key agreement scheme for fog-based smart grid, *Cluster Comput.* 25 (1) (2022) 451–468.
- [7] W. Wang, Q. Chen, Z. Yin, G. Srivastava, T.R. Gadekallu, F. Alsolami, C. Su, Blockchain and PUF-based lightweight authentication protocol for wireless medical sensor networks, *IEEE Internet Things J.* 9 (11) (2021) 8883–8891.
- [8] K.N. Griggs, O. Ossipova, C.P. Kohlios, A.N. Baccarini, E.A. Howson, T. Hayajneh, Healthcare blockchain system using smart contracts for secure automated remote patient monitoring, *J. Med. Syst.* 42 (7) (2018) 1–7.
- [9] D.C. Nguyen, P.N. Pathirana, M. Ding, A. Seneviratne, BEdgeHealth: A decentralized architecture for edge-based IoMT networks using blockchain, *IEEE Internet Things J.* 8 (14) (2021) 11743–11757.
- [10] P.K. Yeng, B. Yang, E.A. Sneekenes, Framework for healthcare security practice analysis, modeling and incentivization, in: 2019 IEEE International Conference on Big Data (Big Data), IEEE, 2019, pp. 3242–3251.
- [11] M. Fotouhi, M. Bayat, A.K. Das, H.A.N. Far, S.M. Pournaghi, M.-A. Doostari, A lightweight and secure two-factor authentication scheme for wireless body area networks in health-care IoT, *Comput. Netw.* 177 (2020) 107333.
- [12] J. Li, Z. Su, D. Guo, K.-K.R. Choo, Y. Ji, PSL-MAAKA: Provably secure and lightweight mutual authentication and key agreement protocol for fully public channels in internet of medical things, *IEEE Internet Things J.* 8 (17) (2021) 13183–13195.
- [13] M. Masud, G.S. Gaba, K. Choudhary, M.S. Hossain, M.F. Alhamid, G. Muhammad, Lightweight and anonymity-preserving user authentication scheme for IoT-based healthcare, *IEEE Internet Things J.* 9 (4) (2021) 2649–2656.
- [14] H. Amintoosi, M. Nikooghadam, M. Shojafar, S. Kumari, M. Alazab, Slight: A lightweight authentication scheme for smart healthcare services, *Comput. Electr. Eng.* 99 (2022) 107803.
- [15] A. Essén, A. Ekholm, Centralization vs. decentralization on the blockchain in a health information exchange context, in: *Digital Transformation and Public Services: Societal Impacts in Sweden and Beyond*, 2020, pp. 58–82.
- [16] S. Yu, Y. Park, A robust authentication protocol for wireless medical sensor networks using blockchain and physically unclonable functions, *IEEE Internet Things J.* 9 (20) (2022) 20214–20228.
- [17] X. Jia, M. Luo, H. Wang, J. Shen, D. He, A blockchain-assisted privacy-aware authentication scheme for internet of medical things, *IEEE Internet Things J.* 9 (21) (2022) 21838–21850.
- [18] A. Tomar, S. Tripathi, BICAV: Blockchain-based certificateless authentication system for vehicular network, *Peer-To-Peer Netw. Appl.* 15 (3) (2022) 1733–1756.
- [19] C. Dobraunig, M. Eichlseder, F. Mendel, M. Schläffer, Ascon v1. 2: Lightweight authenticated encryption and hashing, *J. Cryptol.* 34 (3) (2021) 1–42.
- [20] C.J. Cremers, The scyther tool: Verification, falsification, and analysis of security protocols, in: *International Conference on Computer Aided Verification*, Springer, 2008, pp. 414–418.
- [21] C.-M. Chen, S. Liu, X. Li, S.H. Islam, A.K. Das, A provably-secure authenticated key agreement protocol for remote patient monitoring IoMT, *J. Syst. Archit.* 136 (2023) 102831.
- [22] W. Wang, H. Huang, F. Xiao, Q. Li, L. Xue, J. Jiang, Computation-transferable authenticated key agreement protocol for smart healthcare, *J. Syst. Archit.* 118 (2021) 102215.
- [23] B. Bera, D. Chattaraj, A.K. Das, Designing secure blockchain-based access control scheme in IoT-enabled internet of drones deployment, *Comput. Commun.* 153 (2020) 229–249.
- [24] A.K. Das, M. Wazid, A.R. Yannam, J.J. Rodrigues, Y. Park, Provably secure ECC-based device access control and key agreement protocol for IoT environment, *IEEE Access* 7 (2019) 55382–55397.
- [25] B. Bera, S. Saha, A.K. Das, A.V. Vasilakos, Designing blockchain-based access control protocol in iot-enabled smart-grid system, *IEEE Internet Things J.* 8 (7) (2020) 5744–5761.