Research Article

An Improved Blockchain-Based Authentication Protocol for IoT Network Management

Mostafa Yavari ¹, Masoumeh Safkhani ¹, Saru Kumari ², Sachin Kumar ³,
and Chien-Ming Chen ⁴

¹Computer Engineering Department, Shahid Rajaei Teacher Training University, Tehran, Iran

²Department of Mathematics, Chaudhary Charan Singh University, Meerut 250004, India

³Department of Computer Science and Engineering, Ajay Kumar Garg Engineering College, Ghaziabad 201009, India

⁴Shandong University of Science and Technology, Qingdao, Shandong, China

Correspondence should be addressed to Chien-Ming Chen; chienming.taiwan@gmail.com

Received 19 July 2020; Revised 19 September 2020; Accepted 5 October 2020; Published 28 October 2020

Academic Editor: Debiao He

Copyright © 2020 Mostafa Yavari et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Communication security between IoT devices is a major concern in this area, and the blockchain has raised hopes that this concern will be addressed. In the blockchain concept, the majority or even all network nodes check the validity and accuracy of exchanged data before accepting and recording them, whether this data is related to financial transactions or measurements of a sensor or an authentication message. In evaluating the validity of an exchanged data, nodes must reach a consensus in order to perform a special action, in which case the opportunity to enter and record transactions and unreliable interactions with the system is significantly reduced. Recently, in order to share and access management of IoT devices information with distributed attitude a new authentication protocol based on blockchain is proposed and it is claimed that this protocol satisfies user privacy preserving and security. However, in this paper, we show that this protocol has security vulnerabilities against secret disclosure, replay, traceability, and Token reuse attacks with the success probability of 1 and constant complexity of also 1. We also proposed an improved blockchain-based authentication protocol (IBCbAP) that has security properties such as secure access management and anonymity. We implemented IBCbAP using JavaScript programming language and Ethereum local blockchain. We also proved IBCbAP's security both informally and formally through the Scyther tool. Our comparisons showed that IBCbAP could provide suitable security along with reasonable cost.

1. Introduction

With the advent of Internet of Things (IoT) technologies, a large number of intelligent devices have been developed and integrated into daily life [1–5]. By increasing number of devices and users, current architecture and the communication protocols (centralized system) cannot give enough response to system requirements such as authentication, authorization, and access management. Although security and privacy are important issues in communication, various solutions have been proposed for security and privacy in Internet of Things (IoT) networks [6–12]. One of the important solutions is use of distributed networks instead of centralized or decentralized networks [13–20]. A new and

powerful distributed system is blockchain [21] which for the first time proposed by Satoshi Nakamoto. Blockchain is a technology comprising few old concepts, such as the ledger and the consensus (the agreement of members of a group on an issue). This technology combines these concepts with the help of a peer-to-peer network to access a distributed database with privacy preserving and security properties. Blockchain has many security features such as integrity, distribution, and tamper-proofing. In a blockchain network, all network members take part in the verification process of information which takes part in the alternative role of the trusted third party (TTP) on the system. It is very difficult to manipulate information because of public surveillance of information. Public surveillance is done by a method called

consensus that means 51% of network members need to collaborate to unauthorized changes in the information. Distributing the role of the TTP among network members has also reduced the likelihood of distributed denial of service (DDoS) attacks. As a result, system security is ensured. In contrast, the processing amount and system messages sent to the members of each node in the network are very high. Also, because of the transparency of all information, privacy preserving becomes difficult and new solutions are needed and these solutions should not impede the consensus process. Another problem is its low scalability because of the fixed and unchangeable data and setting. Therefore, the cost of changing some parts of the blockchain network is much more than building a new network. As a result, the cost of upgrading the system is very high.

In this regard, Cha et al. in [22] have presented an authentication scheme for the IoT using blockchain, claiming that their protocol has made it possible for users to access and manage IoT device information with privacy. In fact, they claimed that their protocol was able to establish user privacy and trust in IoT applications. In this paper, we show that the protocol designed by Cha et al. has drawbacks that make their design vulnerable to various attacks such as secret disclosure, traceability, replay, and reuse of Token attacks, which leads to loss of privacy and trust. Security analysis of the Cha et al.'s protocol before use prevents a lot of possible damage. It also makes designers aware of such errors in protocol design and avoid repeating them in their designs. In this paper, we also offer suggestions for troubleshooting this protocol. Using these suggestions, we provide an improved version of the protocol in terms of security and cost, which is called IBCbAP.

1.1. Main Contribution. The contributions of the paper are as follows:

- (1) Presenting security weaknesses of Cha et al. blockchain-based authentication and secure management of privacy preferences scheme.
- (2) Addressing security pitfalls of Cha et al.'s scheme and proposing a new improved one called IBCbAP.
- (3) Formal and informal security analysis of IBCbAP. Formal proofing is done using the Scyther tool.
- (4) Implementation of IBCbAP through JavaScript language and Ethereum local blockchain and considering its functionality and correctness.

1.2. Paper Organization. We organize the rest of the paper as follows. In Section 2, we will look at related works on the blending of two technologies of blockchain and IoT and also provide some explanations on their security requirements and existing issues. We review the Cha et al.'s blockchain-based authentication protocol and describe its security pitfalls in Sections 3 and 4, respectively. In Section 5, we propose a new improved protocol called IBCbAP and describe its security and functionality features. We analyze the security and functionality of IBCbAP in Sections 6 and 7,

respectively. In Section 8, we compare the proposed protocol in terms of the type of blockchain used, the implementation platform, and security features with other related authentication schemes in this area, and at last Section 9 concludes the paper with concluding remarks and suggestions for future works.

2. Related Works and Preliminaries

This section addresses the important security requirements for access control systems in the IoT networks and briefly presents a few related works. Since we used the blockchain to store critical information securely in our proposed protocol, the definition of blockchain technology and how it works are also explained in this section. Also, we briefly explain the smart contract and its usage in our proposed protocol. Finally, we will talk about how the blockchain was used in the related works.

2.1. Blockchain. Blockchain was proposed in 2008 by Satoshi Nakamoto [21]. Blockchain includes blocks where blocks are interconnected like a chain. Each block contains information such as block number, the hash of the previous block, a nonce, and transaction information. By storing the hash of the previous block in each block, the chain will be formed. This chain is called the ledger. Figure 1 shows a simple example of the blockchain ledger. Each node in the network has its ledger. Blockchain uses consensus mechanisms [23, 24] to verify the transaction and update the entire ledger. At the time of adding a new transaction in the ledger, all nodes in the network will check the correctness of the information and, after approving, will add the new transaction to their ledger. Each user subscribes to the network by registering a pair of public and private keys on the network. This is done by recording a transaction. Each user's keys are stored in their wallet. Miners created the blocks. Miners are nodes in the network, tasked with generating and approving blocks to the blockchain. To generate a block, the corresponding node solves a difficult problem, and the one who solves the problem sooner registers its block in the blockchain. Changing an approved block in the ledger is costly and difficult. There are two types of blockchain: public and private. In a public blockchain, anyone can participate in the block generation and consensus process, but in a private blockchain, only preapproved nodes can do this operation. Bitcoin [21] and Ethereum [25] are examples of the public type, and Hyperledger [26] is an example of the private type blockchain.

2.2. Smart Contract. Smart contracts are executable codes and memories, which are stored as transactions in the blockchain. Smart contracts are executed by miners with a fixed cost. By knowing the transaction address, it is possible to call the smart contract for the network members. Because blockchain is irremovable, smart contracts can provide a great deal of confidence. One of the famous blockchains, which is a smart contract provider, is Ethereum [25]. In the Ethereum network, any member can create a smart contract

FIGURE 1: Chain of blocks in the blockchain technology [21].

and share it with anyone. In this paper, we use the testRPC [27] to implement the blockchain network. This library creates a local Ethereum blockchain network.

2.3. Related Work. In the following, we introduce several related kinds of research which have tried to use blockchain's security features, such as distribution, integrity, and tamper-proof properties to create security and privacy in communications protocols. The first blockchain is provided by Bitcoin [21], which stores financial transactions about coins. In Bitcoin, if the transactions change, it can store identity information and access policies. Ouaddah et al. designed a system to maintain privacy and security in a distributed manner, which is their main idea [28]. A problem with Ouaddah et al.'s design is the high computational cost for network members. Novo [29], by decreasing of distribution, has attempted the computations to fit the resources of the devices. Novo separated the blockchain network from the general network and communicates with the devices using management hubs. This solution boosts performance and reduces distribution. Compared to Ouaddah et al.'s design, it boosts computational power to devices and increases the probability of distributed denial of service (DDoS) attacks. In Novo's protocol, miners cannot be a member of the network just cooperating to the blockchain network. The high computational load results from the block generation process. Using the lighter block generation process, we can efficiently use the blockchain network in the Internet of Things (IoT). Dorri et al. proposed a method for the block generation process and consensus, which is based on the limitation of the number of blocks generated per unit time per node [30]. Dorri et al. have claimed that the proposed method is scalable and provides a proportional computational burden to network members. In their scheme, miners are members of the network and their number and membership are changed based on few parameters. Dorri et al. used smart homes network scenario. The interior management of each home is done by a central point inside the house, and it is described in [31]. Lin et al. by using blockchain, ABS (Attribute-Based Signature), and MRE (Multi-Receiver Encryption) have proposed a system for managing industrial devices and data collection [32]. In fact, Lin et al.'s protocol is presented for the development of industry V.4 by blockchain. The structure of Lin et al.'s protocol has five entities, including terminals, blockchain, cloud, industrial network, and physical resources. In this scheme, commands are stored by the terminals at blockchain. The cloud and industrial network monitor the blockchain and find the commands and then perform the

operations. Lin et al. have used distributed reductions to adapt their protocol to IoT conditions.

Hammi et al. have proposed a system to authenticate and authorize IoT devices that also meet security requirements such as confidentiality and integrity. They based the main idea behind their scheme on the use of blockchain security benefits and the segmentation of system components into unique areas. In these areas, known as bubbles, all members are allowed to communicate with their bubble members and identify other bubble members as attackers. ECDLP (Elliptic Curve Discrete Logarithm Problem) based cryptography is used by Hammi et al. This system has appeared in highly practical results, and the authors emphasized that they have achieved very satisfactory results. Xie et al. in [33] used the blockchain to improve security and privacy on the Internet of Vehicles (IoV) and transportation systems using 5G-VANET [34]. In Xie et al.'s scheme, all RSU (Road Side Unit-5G-Station-Vehicle) members are connected and controlled by a central SDN (Software Defined Network) controller. The number of messages sent by any vehicle and the message authorization status are stored in the blockchain. In this scheme, transactions are unchangeable and the vehicles cannot deny sending. As a result, as the error coefficient of a vehicle increases, the miners define more restrictions on the transmission of the vehicle transaction, which will increase system's security. In fact, in their scheme vehicles with data encryption protect privacy. Huang et al. have introduced a system for controlling access to device resources according to device policies in [35]. The overall structure of Huang et al.'s scheme is very similar to the overall structure of Cha et al.'s scheme. Huang et al. used DAG (Directed Acyclic Graph)-based blockchains [36]. DAG-based blockchains are suitable for IoT applications because of their lightness and high speed. Also, Huang et al. have proposed a consensus method. In their proposed consensus method, the nodes have a credit value which comprises two variables: node activity and node honesty. The difficulty of the problem solved in block generation is inversely related to the credit value. The speed of storing a device's transactions depends on its credit value. By reducing the credit value of a malicious device, the cost and time of attack increase. Yao et al. introduced a lightweight blockchain-based authentication system in [37]. Yao et al. specifically recommends this system for using in distributed vehicular fog servers. Yao et al. have enabled one-time authentication to access the services. They used the blockchain to store authentication history and reauthentication is selected by the devices. Sidorov et al. have proposed a blockchain-based supply chain system in [38]. They have used RFID tags to track products and also have used blockchain as a trusted third party. Sidorov et al. have claimed that they created a high level of privacy and security. The used blockchain type in the Sidorov et al.'s system is private and messages are encrypted only using rotation operation, XOR (Exclusive OR) operation, and HW (Hamming Weight) function. Dwivedi et al. have provided a blockchain-based healthcare system in [39]. The need for privacy preserving and distribution in healthcare systems is justifiable reasons for using blockchain in this plan. Dwivedi et al.'s scheme has used cloud and smart contracts, which are used to store patient

data and analyze data according to the instructions of the healthcare provider. They used blockchain to secure the information in the cloud and keep track of changes. In Dwivedi et al.'s scheme, the patient sends information to the smart contract by wearing and using IoT devices and the smart contract decides according to the healthcare provider orders and notifies the relevant doctor.

As seen in this section, many attempts have been made to integrate blockchain technology with the IoT. All of these efforts lead to the growth and maturity of blockchain use in the Internet of Things [40–43]. An example is Cha et al.'s scheme [22], in which the authentication of IoT devices is done through blockchain. In this paper, we show that Cha et al.'s designed structure has its drawbacks and we offer solutions to address them.

3. Cha et al.'s Blockchain-Based Protocol

In this section, we explain Cha et al.'s blockchain-based protocol [22] and its security analysis including secret disclosure, replay, traceability, and reuse Token attacks.

3.1. Cha et al.'s Protocol. Recently, in [22], Cha et al. proposed a new authentication protocol based on blockchain for share and access management of IoT device information as a distributed attitude. In this protocol, there are three main entities called device, user, and blockchain connected gateway (BCG). Devices are the nodes in the network that share information or resources with conditions called Device policy, and other users, device, or people who want to use the information and resources of the devices. BCG is an intermediary between the device and the user. These gateways assess users' eligibility (permissions and authentication) to use the information and resources of the devices according to policies.

We use the scenario described in [22] to explain communication and process sequencing. In this scenario, each user joins the blockchain network and registers its public and private key pair. The BCGs and administrator of devices deploy their smart contract in the blockchain network. Authentication processes and access policies management are done by smart contracts. In this scenario, each device is connected to a BCG. This connection is a logical connection and saved in the device and in the BCG smart contracts. When a device connects to a BCG, it can be said that access to device information is done by this BCG.

As shown in Figure 2, the user first finds the BCG smart contract address and then accesses the list of subset devices. To use any device, the user must declare their agreement with the device's privacy policies. This agreement is stored in the blockchain network so that the BCG can be used at the time of request for access to device information by the user.

3.1.1. Smart Contracts. Here, we explain the implicit smart contract and their obligations in Cha et al.'s authentication protocol. Interactions between the device and the BCG (logical communication), control of device information and privacy policies and BCG management are all done through smart contracts. As shown in Figure 3, the communication between

the devices and the BCG is recorded in the blockchain and is done through it. We can say the blockchain plays the role of a trusted third party (TTP). Therefore, there is no possibility of manipulating or violating information for the protocol's parties.

3.1.2. Cha et al.'s Proposed Signature. The signature used in [22] is based on ECDLP (Elliptic Curve Discrete Logarithm Problem) and has six steps including SETUP, SET-PARTIAL-PRIVATE-KEY, SET-SECRET-VALUE, SET-PUBLIC-KEY, SIGN, and VERIFY. These steps by using notations represented in Table 1 will be briefly discussed in the following subsections and also are shown in Figure 4.

(1) *Setup.* In the first step, according to the secret value of k , the BCG selects two groups of G_1 and G_2 with the same q prime order. The relationship between G_1 and G_2 is $e: G_1^* G_1 \rightarrow G_2$, where P is the generator of G_1 . G_1 has bilinear pairing attribute, so the following relationship exists:

$$\forall m, n \in Z_q^*, \forall S, P \in G_1: e(mS, nP) = e(S, P)^{mn}. \quad (1)$$

The BCG selects its private and public key pair as $s \in Z_q^*$ and $PK_{BCG} = sP$, respectively. Also BCG chooses three hash functions with definitions $H_1: \{0, 1\}^* \times G_1 \rightarrow Z_q^*$, $H_2: \{0, 1\}^* \times G_1 \times G_1 \rightarrow Z_q^*$ and $H_3: \{0, 1\}^* \times G_1 \times G_1 \times G_1 \rightarrow Z_q^*$, and ultimately the values of $G_1, G_2, q, e, P, PK_{BCG}, H_1, H_2, H_3, e(P, P)$ are published by the BCG.

(2) *Set-Partial-Private-Key.* In this step, the BCG calculates R_i, h_i, s_i , and σ_{i_1} values as below:

$$\begin{aligned} R_i &= r_i \cdot P, \\ h_i &= H_1(ID_i, R_i, PK_{BCG}), \\ s_i &= r_i + h_i \cdot s \bmod q, \\ \sigma_{i_1} &= s_i^{-1} \cdot P. \end{aligned} \quad (2)$$

In the above calculations, a random number $r_i \in Z_q^*$ is used which is generated using public parameters, master private key, and the user U_i 's identity ID_i . After that, BCG sends (R_i, σ_{i_1}) to the user; the user once receives the messages immediately checks $e(\sigma_{i_1}, R_i + h_i \cdot PK_{BCG}) \stackrel{?}{=} e(P, P)$, where $h_i = H_1(ID_i, R_i, PK_{BCG})$.

(3) *Set-Secret-Value.* In this step, the user selects a random number as his private key. This key is notated with $x_i \in Z_q^*$.

(4) *Set-Public-Key.* In this step, the user using his secret key, i.e., x_i , generates his public key as $PK_i = x_i \cdot P$.

(5) *Sign.* At the beginning of this step, the user calculates the amount of $k_i = H_2(ID_i, PK_i, R_i, PK_{BCG}, m)$, where m is the message to be signed, and calculates $\sigma_{i_2} = (k_i \cdot s_i + x_i)^{-1} \cdot P$. At the end, the user sends (m, R_i, σ_{i_2}) to the verifier.

(6) *Verify.* In this step, the verifier computes the value of h_i as $H_1(ID_i, R_i, PK_{BCG})$ using the values of PK_i and received (m, R_i, σ_{i_2}) . After that, the verifier calculates k_i as

FIGURE 2: Overview of Cha et al.'s blockchain-based authentication protocol.

FIGURE 3: Device binding communications in Cha et al.'s blockchain-based authentication protocol.

$H_2(ID_i, PK_i, R_i, PK_{BCG}, m)$ and verifies the correctness of signature by checking whether $e(\sigma_i, k_i, (R_i + h_i \cdot PK_{BCG}) + PK_i) \stackrel{?}{=} e(P, P)$. If the condition is fulfilled, the signer is authentic.

3.1.3. Device Binding. The user sends a consent of the device policies (PP_i) to the BCG. Each BCG has a root key. The user's public key (PK_i) is also specified. The BCG initially

generates a random number (nonce N_i) and then, using this value, the public key of the user, BCG root key, and a hash function (H), generates a key ($K_{U_{ip_j}}^{N_i}$). The combination of user consent, policy (P_j), and timestamp (T_{ij}) are encrypted with the generated key and then stored in the blockchain. The BCG sends the transaction ID (TID_{ij}) and the generated key to the user because the user is sure of its accuracy. In future communications, stored information is the basis for

TABLE 1: Summary of notations used in this paper.

Notation	Description
q	Prime order
G_1, G_2	Cyclic groups with the same prime order q
e	Bilinear pairing map $G_1 * G_1 \rightarrow G_2$
P	Generator of G_1
s	Master private key
PK_{BCG}	BCG public key, i.e., $(s.P)$
H, H_1	Hash functions
H_2, H_3	
ID_i	User U_i 's identity
SK_{dbcg}	Shared key between the device and the BCG
m	Message
$E_K(.)$	Symmetric encryption function
P_j	Device policy
PP_j	Consent of the P_j policy
T_{ij}	Timestamps
$K_{U_i}^{N_i}$	Generated key between BCG and user U_i
$PK_i^{P_j}$	User's public key
r_u, r_d, r_b	Random numbers generated by user, device, and BCG, respectively
TID_{ij}	Transaction address added to blockchain because of acceptance policy P_j by U_i user
N_i	Random numbers
SK_{ubcg}	Shared key between the user and the BCG
r_i	Random numbers
$\parallel$	Concatenation

FIGURE 4: Cha et al.'s proposed signature.

licensing access to the device. Figure 5 illustrates the device binding of Cha et al.'s protocol.

3.1.4. Access to the Device. Figure 6 illustrates how the user is accessing the device. The implications of Figure 6 will be discussed in the following.

- The user generates a signature for the access request message (PP_j) using $s_i = r_i + h_i.s$, $R_i = r_i.P$, $h_i = H_1(ID_i, R_i, PK_{BCG})$, $x_i, PK_i, PP_j, K_{ij} = H_2(ID_i, PK_i, R_i, PK_{BCG}, PP_j)$ as $\sigma_{ij} = (k_{ij}.s_i + x_i)^{-1}$ and sends (PP_j, R_i, σ_{ij}) to the BCG.

Once the BCG receives the message, it checks the authenticity of the signature by checking whether $e(\sigma_{ij}, k_{ij}.(R_i + h_i.PK_{BCG}) + PK_i) = e(P, P)$ or not. If

so, the BCG generates a random value ($r_{ij} \in Z_q^*$) and then calculates $R_{ij} = r_{ij}.P$, $l_{ij} = H_1(ID_i, R_{ij}, \sigma_{ij}, PK_{BCG})$, $s_{ij} = r_{ij} + l_{ij}$, $\sigma_{BCG} = s_{ij}^{-1}.P$ and at last the BCG sends (R_{ij}, σ_{BCG}) to the user in response.

After receiving (R_{ij}, σ_{BCG}) from the BCG, the user verifies the validity of the received values by computing $l_{ij} = H_1(ID_i, R_{ij}, \sigma_{ij}, PK_{BCG})$ and checking whether $e(\sigma_{BCG}, R_{ij} + l_{ij}.PK_{BCG}) \stackrel{?}{=} e(P, P)$ or not. If the accuracy of the received information is verified, the access Token is calculated as below:

$$\text{Token} = (PP_j, R_i, \sigma_{ij}, R_{ij}, \sigma_{BCG}). \quad (3)$$

FIGURE 5: Cha et al.'s protocol: device binding phase.

FIGURE 6: Cha et al.'s protocol: access to device phase.

Finally, the user may use the device by sending Token to the device. Access permission depends on the result of verifying the validity of the received Token by the device.

4. Security Analysis of Cha et al.'s Blockchain-Based Authentication Protocol

In this section, we will investigate Cha et al.'s protocol's security. This analysis includes the presentation of security attacks. We have applied four attacks on Cha et al.'s protocol comprising secret disclosure, replay, traceability, and reuse Token attacks. This section describes how the attacks are performed, their success probability and complexity, and a solution to prevent the occurrence of the attacks. We base the proposed protocol on these solutions.

4.1. The Adversary Model. Here, we used the common adversary model [44–48] in which the adversary can fully control and eavesdrop the public communication channel.

So, he/she can eavesdrop, replay, modify, and delete any message or fabricate a new message.

4.2. Secret Disclosure Attack. According to (3), the Token is generated by the values of the sent messages. Since Cha et al. do not consider the channel secure, the attacker can eavesdrop the transferred messages through the channel. As a result, generating Token by the attacker is only possible through the eavesdropping of the channel, resulting in threats to the device's resources and security. For this attack, the attacker proceeds as below:

- (1) The attacker eavesdrops transferred messages for one run of the access to device phase of Cha et al.'s protocol
- (2) Using $PP_j, R_i, \sigma_{ij}, R_{ij}$, and σ_{BCG} which eavesdropped in access to device phase of Cha et al.'s protocol, the attacker can calculate Token as $(PP_j, R_j, \sigma_{ij}, \sigma_{BCG})$
- (3) Using calculated Token in the previous step, the attacker can access and use the intended IoT device

The success probability of the attack is 1 and its complexity is one run of the protocol. We will fix this weakness in our proposed protocol, i.e., IBCbAP. To solve this problem, we do signature using HMAC (Hash-Based Message Authentication Code) function and ECDLP (Elliptic Curve Discrete Logarithm Problem) encryption. Their details will be explained in Section 5.2.

4.3. Replay Attack. Because of the absence of new random values in each session, messages from each session are not different from other sessions, so a message is acceptable in all sessions. The attacker can gain BCG approval by sending messages sniffed in previous sessions and impersonating the user. The steps of this attack are as follows:

- (1) The attacker eavesdrops one session and stores (PP_j, R_i, σ_{ij}) message as N'
- (2) The attacker starts a new session with BCG by sending N' to the BCG
- (3) BCG checks correctness of N' and because of the mentioned weakness in Cha et al.'s protocol, BCG confirms the authenticity of the information
- (4) BCG sends (R_{ij}, σ_{BCG}) to the attacker
- (5) The attacker accesses the device.

This attack is possible with the success probability of one and complexity of only one run of protocol's eavesdropping.

4.4. Traceability Attack. Because of the absence of nonces or timestamp in Cha et al.'s protocol's transferred messages, all of them are static, so it is possible for the attacker to trace the protocol parties using the content of the transferred messages. For this attack, it is enough for the attacker to proceed as below:

- (1) The attacker eavesdrops one run of protocol and stores the transferred messages
- (2) The attacker gets R_i related to U_i user, notated R'_i
- (3) After that, if the attacker sniffs the protocol messages and compares the R_i with R'_i , he/she can determine whether the message is for the U_i user or not, so recognizing the user

The attacker can detect a user, with the probability of 1 and the complexity of eavesdropping one run of the protocol. This attack is possible because in Cha et al.'s protocol, the $R_i = r_i \cdot P$ is calculated only once and does not change in each session. By adding random nonces or timestamp to messages which are generated by all protocol parties, messages sent in each session are different, and the possibility of tracing users is vanishing.

4.5. Reuse Token Attack. Cha et al. in [22] did not explain their method for verifying the Token (in (3)) by the device. Also, despite the existing parameters in the protocol messages, the device cannot ensure that the Token has not duplicated and is correct. In other words, the user or attacker by receiving a Token can use the device's resources

unlimitedly, and the BCG or device cannot detect this situation. This is because no fresh variable (such as random numbers or timestamps) is used in the Token and the device has no role in creating the Token. So, by getting a Token, the attacker and a malicious user can use the Token repeatedly and any members of the network do not figure out. This issue causes the abuse of device resources and violates the claim made by Cha et al. to manage privacy policies. We will solve this problem by using random numbers in each Token separately which are generated by all members who take part in the network. In the next section, we will address all of abovementioned vulnerabilities which lead to proposing an improved version of Cha et al.'s blockchain-based authentication protocol called IBCbAP. We also will conduct formal security analysis and informal security analysis of IBCbAP. Formal proof will be done using the Scyther tool. To show the functionality of IBCbAP, we will implement it by JavaScript language and the Ethereum local blockchain network.

5. IBCbAP: The Improved Protocol

Here, we improve Cha et al.'s protocol's weaknesses by using the HMAC function instead of Cha et al.'s proposed signature. We named the proposed protocol IBCbAP (Improved Blockchain-based Authentication Protocol). According to [49], the speed and memory consumption in an ECDLP (Elliptic Curve Discrete Logarithm Problem) encryption are less than RSA. Therefore, we use the ECDLP encryption method. We do the initial setting and selection of constant values just like Cha et al.'s protocol. As shown in Figures 7 and 8, respectively, the steps of IBCbAP are as below.

- (i) Confirm access policy phase: In this phase, the user, after finding device policy by associated BCG, sends its consent to device policies to the BCG. Also, the BCG generates a SK_{ubcga} secret key and shares with the user. In this phase, the used channel is secure.
- (ii) Create access Token phase: In this phase, the BCG creates a Token for the user for using the device. Token is generated by the user, the device, and the BCG participants securely. The key used in this phase is generated in the Confirm access policy phase in which the used channel is secure.

5.1. Confirm Access Policy. In the first step, the user needs to find the device access policies (P_j) and agree to them. To accept the device policy, he/she sends a message (PP_j) to the BCG according to the policy. The BCG generates a nonce (N_i) upon which it receives PP_j . The shared key between the device and the BCG (SK_{ubcg}) is created using the equation below:

$$SK_{ubcg} = H(ID_i \| N_i \| SK_{BCG}). \quad (4)$$

This key is used in subsequent messages where H is a hash function, and SK_{BCG} is a secret value associated with the BCG. The P_j , PP_j , and a timestamp (T) are encrypted using the

FIGURE 7: IBCbAP: Confirm access policy phase.

FIGURE 8: IBCbAP: Create access Token phase.

generated key and stored as a transaction in the blockchain. The transaction address (TID_{ij}) along with the (SK_{ubcg}) is sent to the user through the secure channel. It is worth noting that the channel between the BCG and the user is secure.

5.2. *Create Access Token.* The user accesses the device through the following steps:

- (i) The user produces $PP_j = E_{SK_{ubcg}}(TID_{ij} || r_u)$, where TID_{ij} , r_u , and SK_{ubcg} are the transaction number associated with accepting device policies, a random number, and a shared secret key between himself and the BCG, respectively. The user sends (PP_j, r_u) to the device.
- (ii) The device once receives the message, generates a random number (r_d), and computes $M_{sig} = \text{HMAC}(r_u || r_d || PP_j, SK_{dbcg})$ and sends (M_{sig}, r_d, r_u, PP_j) to the BCG.
- (iii) Upon receipt of the message, the BCG generates a signature ($M'_{sig} = \text{HMAC}(r_u || r_d || PP_j, SK_{dbcg})$) and compares it with the received signature (M_{sig}). The continuation of the protocol is related to the

satisfying $M'_{sig} = M_{sig}$ condition. BCG decrypts the PP_j and ensures the user's claim using the blockchain network. In the next step, the BCG generates another random number (r_b) and the value of $M'_{sig} = \text{HMAC}(r_b || r_u || TID_{ij}, SK_{ubcg})$. And finally, it sends (M'_{sig}, r_b) to the user.

- (iv) Once the user received the (M'_{sig}, r_b) message, it verifies the authority of the BCG. This is accomplished by checking whether $M'_{sig} \stackrel{?}{=} \text{HMAC}(r_b || r_u || TID_{ij}, SK_{ubcg})$ or not. The device already knows the value of r_u , TID_{ij} , and SK_{ubcg} . The continuation of the protocol is conditional upon the generated signature of the user and the signature from the BCG. In the following, the user generates a signed message as $N_{sig} = \text{HMAC}(r_b, SK_{ubcg})$ and sends it to the BCG.
- (v) The BCG checks whether $N_{sig} \stackrel{?}{=} N'_{sig}$, where $N'_{sig} = \text{HMAC}(r_b, SK_{ubcg})$. If so, it creates access Token and computes $E_{SK_{ubcg}}(\text{Token} || r_u)$ and sends it to the user. Also, it computes $E_{SK_{dbcg}}(\text{Token} || r_d)$ and sends it to the device.

When the user receives the message $E_{SK_{ubcg}}(\text{Token}||tr_u)$ and decrypts it with the key SK_{ubcg} , if obtained r_u is equal to its one, the user trusts the received Token. The same is true for the device.

6. IBCbAP Security Analysis

In this section, we consider the security of IBCbAP informally and also formally. We perform the formal security checking using the Scyther tool [50].

6.1. Informal Security Analysis. Here, we informally prove that the improved protocol can resist against different attacks. Informal analysis is based on the knowledge and creativity of the analyst.

6.1.1. Resistance to DDoS Attack. Because of using blockchain, the architecture of the system moves toward distribution architecture. The distribution of a system reduces the probability of the DDoS attack. However, because of the lack of participation of IoT devices in the blockchain network and IoT devices connected to the blockchain network by BCG, the distribution of the system decreases. IoT devices cannot directly participate in blockchain network, because many resources are needed for this purpose. Due to the use of blockchain as the trusted third party (TTP) and the distribution of IoT devices between different BCGs, the possibility of a DDoS attack on the system is reduced. Also, since keys are not changed at each session and the existence of different keys per user for each device, the possibility of changing values and system desynchronization is eliminated.

6.1.2. Resistance to Replay Attack. Since, in the proposed protocol, all members of the network participate in the Create access Token phase, the possibility of the replay attack is eliminated. Existence of shared secret values (SK_{ubcg}, SK_{dbcg}) between protocol parties and use of freshly random values (r_u, r_d, r_b) by protocol parties lead each session's messages to be different from previous sessions messages; as a result, the possibility of the replay attack vanishes.

6.1.3. Resistance to Secret Disclosure Attack. Important messages sent between the user and the BCG are encrypted by a shared key, such as Token which is encrypted as $E_{SK_{ubcg}}(\text{Token}||tr_u)$. We also encrypt important messages sent between the device and the BCG, such as Token, using a shared secret value, i.e., $E_{SK_{dbcg}}(\text{Token}||tr_d)$. Due to the encryption of messages in the session, important information remains confidential. Also, the existence of shared secret values allows the protocol members to generate the message authentication code such as M_{sig} and N_{sig} , so they can authenticate each other and avoid sending the data to an unauthorized party.

6.1.4. Resistance to Traceability Attack. As mentioned before, because of the involvement of random numbers in all transferred messages and changing these random numbers in each session, the attacker cannot find any fixed value. As a result, the attacker is not capable of tracing protocol's parties using protocol's transferred messages. Also, it is not possible for the attacker to reveal any data in the current session, by eavesdropping the previous transferred messages because of the lack of correlation between messages sent in each session.

6.1.5. Unlinkability of Device. In the proposed protocol, we issue a separate key for each device. Each user is required to process, accept policies phase, and create an access Token phase to use each device independently. We can say that the user uses a separate Token for each device, so with the disclosure of a Token, other devices are not threatened.

6.2. Formal Security Analysis. We have used the Scyther tool to check out the formal security of our proposed blockchain-based authentication protocol. The Scyther is created with Python [51] language and works according to security claims. Claim events are used in role specifications to model intended security properties [52]. There are several predefined claim types in the Scyther tool which are represented in Table 2.

We have used Nisynch, Niagree, and Secret claims. The Nisynch claim checks desynchronization possibility. Niagree claim considers that an agreement will not commit to the changed values between the parties and the Secret claim states that the amount referred to it is secret. IBCbAP written in the Scyther language, i.e., security protocol description language (spdl) with three roles, i.e., user, device, and blockchain connected gateway, and security claims along with the output of the Scyther tool for its security verification are shown in Figures 9 and 10, respectively.

As you can see in Figure 10, the Scyther tool could not find an attack for IBCbAP. Therefore, IBCbAP provides a good level of security.

7. IBCbAP Implementation

We have implemented IBCbAP to test the functionality and the feasibility of implementation with existing technologies. We have used JavaScript in the form of Nodejs [53] technology to simulate the proposed protocol. The reason for this choice is WEB3.js [54] library to contact blockchain. This library uses the RPC (Remote Procedure Call) protocol to interact with blockchain nodes. The system used in this simulation includes features such as 4 GB RAM, 1 TB HDD, and Core i7 2.47 GHz CPU.

7.1. Confirm Access Policy. The implemented architecture has four participants (user, BCG, device, and blockchain network). User to use P_j device sends the PP_j to the BCG and then receives the shared key between itself and BCG (SK_{ubcg}) through a secure channel. How to generate SK_{ubcg} is

TABLE 2: Claim events which defined in the Scyther tool.

Claim	Description
Secret	There is no unauthorized access to the parameters transmitted in the protocol. For example, $\text{claim}(\text{Initiator}, \text{Secret}, n)$ means that the value of n remains confidential in the sense of initiator, and the third party has not been able to access it.
Alive	$\text{claim}(R, \text{alive}, R')$ means that R' is always on the line and is linked to R . In fact, the presence and survival of R' are requested by R .
Weakagree	It ensures that the receiver has received message at the time that was sent by the sender. There is no agreement on the data transmitted.
Niagree	$\text{claim}(R, \text{Niagree}, R')$ claims that the agreed values between the receiver and the transmitter will remain unchanged.
Nisynch	$\text{claim}(R, \text{Nisynch}, R')$ claims that each message sent by the sender (R) corresponds to receiving the message by the receiver, and all the steps are executed correctly and without interruption.

```

usertype sessionkey;
usertype integer;

protocol bcggateway(device,user,bcg)
{
  hashfunction h;
  role user
  {
    fresh ru : Nonce;
    macro tid= h(ru);
    var token: sessionkey;
    var msig'';
    var rb,rb' :Nonce;
    macro pp={ru,tid}k(user,bcg);
    macro nsig={rb}k(user,bcg);

    send_1(user,device, pp,ru);
    recv_3(bcg,user,rb,msig'');
    match(msig'',{rb,ru,tid}k(user,bcg));
    send_4(user, bcg, nsig);
    recv_6(bcg,user,{token,rb'}k(user,bcg));

    claim_u1(user,Secret,token);
  }

  claim_u2(user,Nisynch);
  claim_u3(user,Niagree);
}

role device
{
  var ru,rb' : Nonce;
  var token: sessionkey;
  var pp;
  fresh rd: Nonce;
  macro msig = {ru,rd,pp}k(device,bcg);

  recv_1(user,device, pp,ru );
  send_2(device,bcg, msig,rd,ru,pp);
  recv_5(bcg ,device,{token,rb'}k(device,bcg));

  claim_d1(device,Secret,token);
  claim_d2(device,Nisynch);
  claim_d3(device,Niagree);
}

role bcg
{
  fresh token: sessionkey;
  var rd,ru: Nonce;

  var tid;
  fresh rb,rb' : Nonce;
  var pp,nsig,msig;
  macro msig''={rb,ru,tid}k(device,bcg);

  recv_2(device,bcg, msig,rd,ru,pp );
  match({rd,ru,pp}k(device,bcg),msig);

  send_3(bcg,user,rb,msig'');
  recv_4(user, bcg,nsig );
  match(nsig,{rb}k(user,bcg));
  send_5(bcg ,device,{token,rd}k(device,bcg));
  send_6(bcg,user,{token,rb'}k(user,bcg));

  claim_b1(bcg,Secret,token);
  claim_b2(bcg,Nisynch);
  claim_b3(bcg,Niagree);
}

```

FIGURE 9: Scyther implementation of IBCbAP in the spdl.

Claim	Status	Comments
bcggateway user bcggateway,u1 Secret token	Ok	No attacks within bounds.
bcggateway,u2 Nisynch	Ok	No attacks within bounds.
bcggateway,u3 Niagree	Ok	No attacks within bounds.
device bcggateway,d1 Secret token	Ok	No attacks within bounds.
bcggateway,d2 Nisynch	Ok	No attacks within bounds.
bcggateway,d3 Niagree	Ok	No attacks within bounds.
bcg bcggateway,b1 Secret token	Ok	No attacks within bounds.
bcggateway,b2 Nisynch	Ok	No attacks within bounds.
bcggateway,b3 Niagree	Ok	No attacks within bounds.

Done.

FIGURE 10: IBCbAP's security analysis result, by the Scyther tool.

explained in Section 5.2. The user uses this key in the create access Token phase. At the Confirm access policy phase, the channel is secure. We run this phase 10 times and took 1363 milliseconds. Most of the time spent in this phase is due to

storing the user information in the blockchain. A view of connecting to the device and confirming the device steps are shown in Figures 11 and 12, respectively.

7.2. Create Access Token. Figure 8 shows the Create access Token phase. At this phase, the user starts the process of issuing the access Token by sending r_u and PP_j to the device. According to what was mentioned in Section 5.2, the device starts the Token creation process when the user receives the message. Finally, BCG creates a Token and sends it as $E_{SK_{ubcg}}(\text{Token} \parallel tr_u)$ to the user and as $E_{SK_{dbcg}}(\text{Token} \parallel tr_d)$ to the device. We performed this phase 10 times and have a total time of 1200 milliseconds. The major time consumed at this phase is related to encryption operations. Figure 13 shows the graphical user interface indicating successful receipt of the Token from BCG.

7.3. Ethereum Network. To implement the blockchain network, we have used the testRPC [27] library. This library creates a local blockchain network. Figure 14 illustrates the launch of this library. The testRPC default creates 10

FIGURE 11: The graphical user interface of implemented IBCbAP: Connect to device.

FIGURE 12: The graphical user interface of implemented IBCbAP: Accept device by the user.

FIGURE 13: The graphical user interface of implemented IBCbAP: successful receipt of the Token from BCG.

```

File Edit View Search Terminal Help
king@saws:~$ testrpc
EthereumJS TestRPC v6.0.3 (ganache-core: 2.0.2)

Available Accounts
=====
(0) 0x6a0a00ccaf3b066897e0b9cd1c9ffffbfa2a8c9b
(1) 0x8686a4122bb479b6fec64f98631403d709d6b497
(2) 0xf6d2932e3c130588ed0a15ac15ed88d61ffb148a
(3) 0x737b956b288c4b0b93233a15c5a6778d58846359
(4) 0x2472784b2b79d533507d2660b3aca748e501a93
(5) 0x31896edeac8c74c3958836dab23d2c27ffe0fd
(6) 0x736f621c1947b72c0e99d9d08041c49d81aeb918
(7) 0x22f0ea96a7513be6891e33b47ee68654d9502dd9
(8) 0x10cdf31e38c5118da3b6d5200286a7248fe9f59
(9) 0xb27f74d9c77c00540e9b75c4c6ebf9c82a8a778a

Private Keys
=====
(0) 8036271b90a5836ce36952c10fe243623a35a1b3cea3f935426eb864729e4a78
(1) a8661f3cc3cb7f49e1d852af15309497e33a9238109fb76cbfa491776b5cae4c
(2) 5bbe2c04d5536d5c4921e0b7c7dcb99622bbf2090d250180d41545204d843d5b
(3) 9e97f4939e4b646e18f8b550017f4b56ced8e2ce795b60788b86ab8acc9c3f30
(4) a465ddb4a52e01ca97ca4f2f22bc0159094f1f564aba3a48a69b7c8131b9551
(5) 3d42cd3a08b5262ca3257430d3ec8bb498f8f2016bb7f6135e0d64e10e5af3e3
(6) 3d2d35486aa61b637a954883da53ef62d704d620f650507f6bc668898c9f59de
(7) 1431e096b6cd8cc19b026f95ff50109f973e6b2bed191b6ee2297095a407f4ca
(8) 8277e682ee8931b5955fd98910bd7803c05dc8b349a446e28ab375415c41c24d
(9) d5f44821060740e8a9234a09acF2b27806495fb18428f590c5cadf5a5550a264

HD Wallet
=====
Mnemonic: ivory bronze wrist humble help distance guitar crew category umbrella release item
Base HD Path: m/44'/60'/0'/0/{account_index}

Listening on localhost:8545

```

FIGURE 14: Creation of Ethereum local network by launching testRPC library.

FIGURE 15: Creating smart contract in IBCbAP.

TABLE 3: Comparison between IBCbAP and related protocols.

	Replay attack resistance	Secret disclosure attack resistance	Blockchain type	Consensus mechanism	Implementation
IBCbAP	✓	✓	Public	Proof of stack	JavaScript, RPC protocol
[22]	×	×	Public	Proof of stack	Java, RPC protocol
[28]	✓	✓	Public	Proof of work	Berkeley DB version 4.8., bitcoind regtest
[29]	✓	✓	Not specified	Not specified	Docker [58] and truffle [59]
[32]	✓	✓	Private	Bayesian fault	JUICE [60]
[61]	✓	✓	Public	Proof of work	C++

accounts when building a local blockchain. We use one of these accounts. We used Remix tool to deploy the initial smart contract of the BCG, and Solidity language wrote the

smart contract (see Figure 15). It is worth noting that the smart contract used is simplified and is suitable for implementing Confirm access policy and Create access

Token phase. We used the Crypto [55] library and the sha256 algorithm to generate HMAC. We used the eccrypto [56] library to perform ECC encryption. Also, we used the Express [57] framework to build the server for each protocol party. It should be noted that we set up the servers for each protocol party and blockchain network on a single system on separate ports. As a result, each phase is short. So, here we have shown it is possible to implement the proposed protocol with existing technologies and facilities. It is worth noting that we achieve measured times with only one device, one user, and one port.

8. Comparison

In this section, we compare our proposed protocol with its predecessor, i.e., Cha et al.'s protocol as well as other related schemes. As can be seen in Table 3, our proposed protocol has complete security compared to other protocols. Since the type of blockchain used and the specifications of the system are very different in different designs, we only got the time in our system and it was not possible to compare time with other designs. The time of our proposed protocol for Create access Token and Confirm access policy phases is 1200 and 1363 milliseconds, respectively.

9. Conclusion

Nowadays, blockchain and Internet of Things (IoT) technologies are bonding. The blockchain first attracted attention as part of a wave of crypto currencies, especially Bitcoin, which challenged the normal course of financial transactions. But it was not the blockchain financial transactions that caught the attention of IoT activists, but rather the data exchanges. Blockchain is essentially an antihacking, distributed, and event logging mechanism that appears to be very useful for solving key issues related to networks where connected devices automatically interact with each other, i.e., IoT. Because of the importance of security in IoT, many schemes have been proposed in this subject. In this paper, we examined Cha et al.'s blockchain-based authentication protocol. We illustrated the weaknesses of Cha et al.'s protocol by applying secret disclosure, replay, traceability, and reuse Token attacks with a success probability of one and complexity of one run of protocol. To address Cha et al.'s protocol's security vulnerabilities, we also proposed an improved protocol called IBCbAP and proved its security in an informal way and also in formal way by the Scyther tool. Finally, we implemented IBCbAP using the JavaScript programming language and the Ethereum local blockchain network. We investigated the feasibility of implementing IBCbAP and measured the timing of some processes. The comparisons show that IBCbAP compared to its predecessor, i.e., Cha et al.'s protocol, is completely secure and the cost and time of its implementation are reasonable. One of the most important issues in the IoT network is the transfer of ownership. In future works, we plan to work on the design of blockchain-based ownership transfer protocols which enables the transfer of ownership in a distributed and secure manner.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

- [1] E. K. Wang, J. Yu, C.-M. Chen, S. Kumari, and J. J. Rodrigues, "Data augmentation for internet of things dialog system," *Mobile Networks and Applications*, pp. 1–14, 2020.
- [2] C.-M. Chen, B. Xiang, Y. Liu, and K.-H. Wang, "A secure authentication protocol for internet of vehicles," *IEEE Access*, vol. 7, pp. 12 047–12 057, 2019.
- [3] K.-H. Wang, C.-M. Chen, W. Fang, and T.-Y. Wu, "On the security of a new ultra-lightweight authentication protocol in iot environment for rfid tags," *The Journal of Supercomputing*, vol. 74, no. 1, pp. 65–70, 2018.
- [4] S. Qu, L. Zhao, and Z. Xiong, "Cross-layer congestion control of wireless sensor networks based on fuzzy sliding mode control," *Neural Computing and Applications*, vol. 32, no. 17, pp. 13505–13520, 2020.
- [5] C.-M. Chen, Y. Huang, K.-H. Wang, S. Kumari, and M.-E. Wu, "A secure authenticated and key exchange scheme for fog computing," *Enterprise Information Systems*, pp. 1–16, 2020.
- [6] C.-T. Li, C.-C. Lee, and C.-Y. Weng, "An extended chaotic maps based user authentication and privacy preserving scheme against dos attacks in pervasive and ubiquitous computing environments," *Nonlinear Dynamics*, vol. 74, no. 4, pp. 1133–1143, 2013.
- [7] C.-C. Lee, M.-S. Hwang, and I.-E. Liao, "Security enhancement on a new authentication scheme with anonymity for wireless environments," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 5, pp. 1683–1687, 2006.
- [8] C.-T. Li, C.-Y. Weng, and C.-C. Lee, "A secure rfid tag authentication protocol with privacy preserving in telecare medicine information system," *Journal of Medical Systems*, vol. 39, no. 8, p. 77, 2015.
- [9] S. K. Dwivedi, R. Amin, and S. Vollala, "Blockchain based secured information sharing protocol in supply chain management system with key distribution mechanism," *Journal of Information Security and Applications*, vol. 54, p. 102554, 2020.
- [10] S. K. Dwivedi, R. Amin, S. Vollala, and R. Chaudhry, "Blockchain-based secured event-information sharing protocol in internet of vehicles for smart cities," *Computers & Electrical Engineering*, vol. 86, p. 106719, 2020.
- [11] A. Irshad, M. Usman, S. A. Chaudhry, H. Naqvi, and M. Shafiq, "A provably secure and efficient authenticated key agreement scheme for energy internet based vehicle-to-grid technology framework," *IEEE Transactions on Industry Applications*, vol. 56, no. 4, pp. 4425–4435, 2020.
- [12] K. Mahmood, J. Arshad, S. A. Chaudhry, and S. Kumari, "An enhanced anonymous identity-based key agreement protocol for smart grid advanced metering infrastructure," *International Journal of Communication Systems*, vol. 32, no. 16, p. e4137, 2019.
- [13] P. K. Sharma, N. Kumar, and J. H. Park, "Blockchain technology toward Green IoT: opportunities and challenges," *IEEE Network*, vol. 34, no. 4, pp. 263–269, 2020.

- [14] V. Dedeoglu, R. Jurdak, A. Dorri et al., "Blockchain technologies for IoT," in *Advanced Applications of Blockchain Technology*, pp. 55–89, Springer, Berlin, Germany, 2020.
- [15] S. Shamshad, K. Minahil, K. Mahmood, and C.-M. Chen, "A secure blockchain-based e-health records storage and sharing scheme," *Journal of Information Security and Applications*, vol. 55, p. 102590, 2020.
- [16] B. C. Florea and D. D. Taralunga, "Blockchain IoT for smart electric vehicles battery management," *Sustainability*, vol. 12, no. 10, p. 3984, 2020.
- [17] G. Yu, X. Zha, X. Wang et al., "Enabling attribute revocation for fine-grained access control in blockchain-IoT systems," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1213–1230, 2020.
- [18] P. P. Ray, D. Dash, K. Salah, and N. Kumar, "Blockchain for IoT-based healthcare: background, consensus, platforms, and use cases," *IEEE Systems Journal*, 2020.
- [19] G. Rathee, M. Balasaraswathi, K. P. Chandran, S. D. Gupta, and C. Boopathi, "A secure IoT sensors communication in industry 4.0 using blockchain technology," *Journal of Ambient Intelligence and Humanized Computing*, 2020.
- [20] S. Gupta, V. Malhotra, and S. N. Singh, "Securing IoT-driven remote healthcare data through blockchain," in *Advances in Data and Information Sciences*, pp. 47–56, Springer, Berlin, Germany, 2020.
- [21] S. Nakamoto, *Bitcoin: A Peer-To-Peer Electronic Cash System*, 2008, <https://bitcoin.org/en/bitcoin-paper>.
- [22] S.-C. Cha, J.-F. Chen, C. Su, and K.-H. Yeh, "A blockchain connected gateway for BLE-based devices in the internet of things," *IEEE Access*, vol. 6, pp. 24 639–724 649, 2018.
- [23] E. K. Wang, R. Sun, C.-M. Chen, Z. Liang, S. Kumari, and M. K. Khan, "Proof of X-repute blockchain consensus protocol for IoT systems," *Computers & Security*, vol. 95, p. 101871, 2020.
- [24] E. K. Wang, Z. Liang, C.-M. Chen, S. Kumari, and M. K. Khan, "PoRX: a reputation incentive scheme for blockchain consensus of IIoT," *Future Generation Computer Systems*, vol. 102, pp. 140–151, 2020.
- [25] V. Buterin, "Ethereum White Paper," GitHub Repository, pp. 22–23, 2013.
- [26] Hyperledger, "Open Source Blockchain Technologies," <https://www.hyperledger.org/>, 2020.
- [27] TestRPC, "Library," <https://www.npmjs.com/package/ethereumjs-testrpc>, 2020.
- [28] A. Ouaddah, A. Abou Elkalam, and A. Ait Ouahman, "Fairaccess: a new blockchain-based access control framework for the internet of things," *Security and Communication Networks*, vol. 9, no. 18, pp. 5943–5964, 2016.
- [29] O. Novo, "Blockchain meets IoT: an architecture for scalable access management in IoT," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, 2018.
- [30] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "LSB: a lightweight scalable blockchain for IoT security and privacy," 2017, <https://arxiv.org/abs/1712.02969v1>.
- [31] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, "Blockchain for IoT security and privacy: the case study of a smart home," in *Proceedings of the 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom workshops)*, pp. 618–623, IEEE, Kona, HI, USA, March 2017.
- [32] C. Lin, D. He, X. Huang, K.-K. R. Choo, and A. V. Vasilakos, "BSeIn: a blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0," *Journal of Network and Computer Applications*, vol. 116, pp. 42–52, 2018.
- [33] L. Xie, Y. Ding, H. Yang, and X. Wang, "Blockchain-based secure and trustworthy internet of things in SDN-enabled 5g-VANETs," *IEEE Access*, vol. 7, pp. 56 656–56 666, 2019.
- [34] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5G for vehicular communications," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 111–117, 2018.
- [35] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards secure industrial IoT: blockchain system with credit-based consensus mechanism," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3680–3689, 2019.
- [36] S. Popov, "The Tangle," Cit. on, p. 131, 2016.
- [37] Y. Yao, X. Chang, J. Misis, V. B. Misis, and L. Li, "BLA: blockchain-assisted lightweight Anonymous authentication for distributed vehicular fog services," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3775–3784, 2019.
- [38] M. Sidorov, M. T. Ong, R. V. Sridharan, J. Nakamura, R. Ohmura, and J. H. Khor, "Ultralightweight mutual authentication RFID protocol for blockchain enabled supply chains," *IEEE Access*, vol. 7, pp. 7273–7285, 2019.
- [39] A. Dwivedi, G. Srivastava, S. Dhar, and R. Singh, "A decentralized privacy-preserving healthcare blockchain for IoT," *Sensors*, vol. 19, no. 2, p. 326, 2019.
- [40] E. K. Wang, C.-M. Chen, D. Zhao, W. H. Ip, and K. L. Yung, "A dynamic trust model in internet of things," *Soft Computing*, vol. 24, no. 8, pp. 5773–5782, 2020.
- [41] H. Xiong, Y. Wu, C. Jin, and S. Kumari, "Efficient and privacy-preserving authentication protocol for heterogeneous systems in IIoT," *IEEE Internet of Things Journal*, 2020.
- [42] H. Zhu, X. Wang, C.-M. Chen, and S. Kumari, "Two novel semi-quantum-reflection protocols applied in connected vehicle systems with blockchain," *Computers & Electrical Engineering*, vol. 86, p. 106714, 2020.
- [43] J.-H. Hsiao, R. Tso, C.-M. Chen, and M.-E. Wu, "Decentralized e-voting systems based on the blockchain technology," in *Advances in Computer Science and Ubiquitous Computing*, pp. 305–309, Springer, Berlin, Germany, 2017.
- [44] S. Hussain and S. A. Chaudhry, "Comments on "biometrics-based privacy-preserving user authentication scheme for cloud-based industrial internet of things deployment,"" *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 936–10 940, 2019.
- [45] K. Mansoor, A. Ghani, S. Chaudhry, S. Shamshirband, S. Ghayyur, and A. Mosavi, "Securing IoT-based RFID systems: a robust authentication protocol using symmetric cryptography," *Sensors*, vol. 19, no. 21, p. 4752, 2019.
- [46] A. Ghani, K. Mansoor, S. Mehmood, S. A. Chaudhry, A. U. Rahman, and M. Najmus Saqib, "Security and key management in IoT-based wireless sensor networks: an authentication protocol using symmetric key," *International Journal of Communication Systems*, vol. 32, no. 16, p. e4139, 2019.
- [47] C.-M. Chen, K.-H. Wang, K.-H. Yeh, B. Xiang, and T.-Y. Wu, "Attacks and solutions on a three-party password-based authenticated key exchange protocol for wireless communications," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 8, pp. 3133–3142, 2019.
- [48] K.-H. Wang, C.-M. Chen, W. Fang, and T.-Y. Wu, "A secure authentication scheme for internet of things," *Pervasive and Mobile Computing*, vol. 42, pp. 15–26, 2017.
- [49] D. Mahto, D. A. Khan, and D. K. Yadav, "Security analysis of elliptic curve cryptography and RSA," in *Proceedings of the World Congress on Engineering*, pp. 419–422, London, UK, July 2016.

- [50] Scyther, “Tool,” <https://people.cispa.io/cas.cremers/scyther>, 2020.
- [51] Python, “Programming Language,” <https://www.python.org/>, 2020.
- [52] C. J. F. Cremers, “The Scyther tool: verification, falsification, and analysis of security protocols,” in *Computer Aided Verification*, pp. 414–418, Springer Berlin Heidelberg, Berlin, Germany, 2008.
- [53] N.js, “Nodejs,” <https://nodejs.org/>, 2020.
- [54] WEB3.js, <https://web3js.readthedocs.io/en/1.0/>, 2020.
- [55] Crypto, “L,” <https://nodejs.org/api/crypto.html/>, 2020.
- [56] Eccrypto, “Library,” <https://www.npmjs.com/package/eccrypto/>, 2020.
- [57] Express, “Framework,” <https://expressjs.com/>, 2020.
- [58] Docker, “Get Started with Docker,” <https://www.docker.com/>, 2020.
- [59] G. N. D’andrea, “Truffle—simple development framework for ethereum,” <https://libraries.io/npm/truffle/2.1.1>, 2020.
- [60] 2020 JUICE, <https://www.juzhen.io/>.
- [61] M. T. Hammi, B. Hammi, P. Bellot, and A. Serhrouchni, “Bubbles of trust: a decentralized blockchain-based authentication system for IoT,” *Computers & Security*, vol. 78, pp. 126–142, 2018.