

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/2023.DOI

# Private Blockchain Envisioned Access Control System for Securing Industrial IoT-Based Pervasive Edge Computing

**SOURAV SAHA<sup>1</sup>, BASUDEB BERA<sup>2</sup>, ASHOK KUMAR DAS<sup>3</sup>, (Senior Member, IEEE), NEERAJ KUMAR<sup>4</sup>, (Senior Member, IEEE), SK HAFIZUL ISLAM<sup>5</sup>, (Senior Member, IEEE), YOUNGHO PARK<sup>6</sup>, (Member, IEEE)**

<sup>1</sup>“Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India”

E-mail: sourav.saha@research.iiit.ac.in

<sup>2</sup>“Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India”

E-mail: basudeb.bera@research.iiit.ac.in

<sup>3</sup>“Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India”

E-mail: iitkgp.akdas@gmail.com, ashok.das@iiit.ac.in

<sup>4</sup>“Department of Computer Science and Engineering, Thapar University, Patiala 147 004, India”

E-mail: neeraj.kumar@thapar.edu

<sup>5</sup>“Department of Computer Science and Engineering, Indian Institute of Information Technology Kalyani, West Bengal 741 235, India”

E-mail: hafi786@gmail.com

<sup>6</sup>“School of Electronics Engineering, Kyungpook National University, Daegu 41566, Republic of Korea”

E-mail: parkyh@knu.ac.kr

Corresponding authors: “Ashok Kumar Das; Youngho Park”

This research was supported by the “Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2020R1I1A3058605).” This work was supported by the “Blockchain project under the Ripple Centre of Excellence (CoE) Scheme, CoE in Blockchain, IIIT Hyderabad, India (No. IIIT/R&D Office/Internal Projects/Ripple/2021-22/009)”. This work was also supported by the “Unified Blockchain Framework for offering National Blockchain Service, Ministry of Electronics & Information Technology, New Delhi, Government of India (No. 4(4)/2021-ITEA).”

**ABSTRACT** The Industrial Internet of Things (IIoT) is able to connect machines, analytics and people with IoT smart devices, gateway nodes and edge devices to create powerful intuitiveness to drive smarter, faster and effective business agreements. IIoT having interconnected machines along with devices can monitor, gather, exchange, and analyze information. Since the communication among the entities in IIoT environment takes place insecurely (for instance, wireless communications and Internet), an intruder can easily tamper with the data. Moreover, physical theft of IoT smart devices provides an intruder to mount impersonation and other attacks. To handle such critical issues, in this work, we design a new private blockchain-envisioned access control scheme for Pervasive Edge Computing (PEC) in IIoT environment, called PBACS-PECIIoT. We consider the private blockchain consisting of the transactions and registration credentials of the entities related to IIoT, because the information is strictly confidential and private. The security of PBACS-PECIIoT is significantly improved due to usage of blockchain as immutability, transparency and decentralization along with protection of various potential attacks. A meticulous comparative analysis exhibits that PBACS-PECIIoT achieves greater security and more functionality features, and requires low costs for communication and computational as compared to other pertinent schemes.

**INDEX TERMS** Industrial Internet of Things (IoT), edge computing, blockchain, access control, key agreement, security.

## I. INTRODUCTION

In recent years, the Internet of Things (IoT) has received a great attention across various fields, particularly in the Industrial Internet of Things (IIoT) environment. The assimilation of IoT and industry plays a significant role in building a large-scale smart automation industry. The IIoT endorse

to provides scalable, cost effective, and secure system for manufacturing industrial goods. Recently, the applications of IIoT systems are escalating in various fields, such as smart manufacturing, transportation, aerospace, energy, logistics, healthcare and so on [1]. In addition, it is expected the market

of IIoT will reach more than \$110 billion USD by 2025 [2]. Recently, Pervasive Edge Computing (PEC) becomes a very emerging computing standard. PEC consists of various heterogeneous mobile edge devices, such as “smartphones”, “tablets”, “IoT smart devices”, “gateway nodes”, “edge devices”, and so on. The devices can then communicate with each other to sense, process the sensing information and also to build various applications at the network edge [3].

The rapid growth of IIoT leads to many security attacks, such as “man-in-the-middle”, “impersonation”, “replay” and “privileged-insider” attacks, which may cause serious damage to the IIoT system. Since most traditional and existing IIoT infrastructures are based on centralized system, they are expensive, inefficient and also vulnerable to a “single-point failure”. Recently, combining blockchain technology and IoT-based security solutions achieves popularity among the researchers. Important features of blockchain (decentralization, tamper-proof, trustworthiness, traceability and immutability) provide more functionalities and greater help to the IIoT systems.

Blockchain is a distributed chain of structure on a decentralized Peer-to-Peer (P2P) network, which eliminates the requirement of centrally controlled system and allows the network entities to store the data in a distributed fashion. Considering the demand of large-scale IIoT systems, it becomes infeasible and inefficient to store the huge volumes of data in a traditional IIoT system. Therefore, we feel that a great essence in designing blockchain-based IIoT system for PEC. Thus, it should provide an efficient and robust solution to deal with the security requirements needed for PEC in IIoT environment. Since the information produced in the IIoT environment is strictly private and confidential, the information must not be leaked in public. Moreover, due to wireless communication happen among different entities in IIoT, an adversary should not be able to tamper with the sensitive data. Tampering of data may include intercepting, modifying, deleting or even inserting fake information during communication.

Integrating IIoT with blockchain technology in order to develop a “secure, distributed, and stable blockchain IIoT network” seems to be a natural way [4]. In fact, the integration of blockchain along with IIoT attracted a lot of interests among the stakeholders across industry and academia as well [5]. Due to drawbacks present in IIoT-based “intelligent manufacturing system (IMS)” and also the challenging problem associated to apply the blockchain in IMS, Zhang *et al.* [6] suggested to combine IIoT with the “permissioned blockchain”. In this regard, they designed an efficient “Manufacturing Blockchain of Things (MBCoT) architecture” for the configuration of a secure, decentralized, and traceable IMS. Zhang *et al.* [7] also proposed a “multi-access edge computing (MEC) enabled framework”. It helps in “data security assurance” and “system latency performance” improvement.

To deal with these issues, we represent a novel access control scheme. It allows secure communication among IoT

smart devices and their relevant gateway nodes through the designed access control process. It also provides secure communication among the gateway nodes and the connected edge servers through the designed key management process. The secure transactions are then transformed into various blocks, and addition of those blocks are done through the “Practical Byzantine Fault Tolerance (PBFT)” introduced by Castro *et al.* [8]. It is done by a group of P2P edge servers network in the private blockchain.

### A. MOTIVATION

In IIoT environment, there are various types of applications connected with the system and they integrate large-scale discrete heterogeneous data. Such data can be from the smart sensor data, health care data, traffic data, environmental monitoring data, and industrial manufacturing data. In some smart energy industries, sensors, machines, and actuators collect huge amount of data such as energy, air quality, fault and resource prediction, and product planning from various locations. It further produces large data and enforce a huge amount of processing time to store the data in traditional centralized system. Moreover, in chemical industry, there is an extensive amount of critical data, such as reactivity of a catalyst in different temperature and air pressure conditions, results after the chemicals reactions. In such scenario, inefficiency of IIoT system can seriously damage the productivity of the industry.

With the help of the cloud computing upgradation, IoT platform can process information in a traditional manner and transform the information into the real time actions. While the cloud storage becomes an important role in an IoT or IIoT environment, however there are issues related to threat of data, transparency and privacy preservation. This demands that we require to integrate the blockchain technology with the industrial IoT applications. Since the blockchain helps in providing the trusted sharing services where the reliable information and data can be retrieved, the data (information) can be then traceable. At the same time, the blockchain is also immutable; thus it enhances the security as well. Therefore, integration of decentralized blockchain in IIoT system can enable better efficiency, transparency and guarantee security solutions.

### B. RELATED WORK

In recent years, “access control and authentication” are widely-used two main security mechanisms in providing security in IoT-enabled environments [9]–[20].

#### 1) Nonblockchain-Based Schemes

Li *et al.* [21] proposed an access control method in an “Industrial Wireless Sensor Network (IWSN)” environment. Their scheme permits a user to authorize, revoke, and authenticate for accessing real time information inside IWSN. Though their protocol supports both public verifiability and ciphertext authenticity, but it is impractical because of heavy

computational overheads due to usage of the costly bilinear pairing operations.

Bilal and Kang [22] also designed an authentication approach in WSN deployment tailored to the IoT environment. In their protocol, a sensor node can establish multiple concurrent sessions to access information securely from other sensor nodes. Unfortunately, their approach is vulnerable to “parallel-session hijacking attack”.

Xue *et al.* [23] suggested an access control mechanism for a smart home environment. Their scheme allows “authentication”, “secure information access”, and “unified storage provision” at the same time. However, the primary drawback related to this approach is that it does not provide “key agreement”. Moreover, their scheme is vulnerable to “Ephemeral Secret Leakage (ESL) attack” under the CK-adversary model mentioned in the threat model (see Section I-B).

Li *et al.* [24] presented a three-factor user authentication scheme for IIoT environment. Unfortunately, their scheme fails to provide forward security and mobile device loss attack. Luo *et al.* [25] designed another access control mechanism for WSN-based IoT environment. Since their scheme is based on the identity based cryptographic technique, it is obviously heavy in computation due to costly bilinear pairing operations.

Li *et al.* [26] designed an elliptic curve cryptography (ECC)-based authentication scheme for IIoT which preserves privacy of the user and gateway nodes, and also provides wrong password detection mechanism quickly. Zeng *et al.* [27] designed an “anonymous user authentication (E-AUA)” protocol for both users and servers in an IoT environment. E-AUA uses multi-server environment to provide better services and also to overcome network congestion. Their scheme is also computationally expensive as costly bilinear pairing operations are applied. Moreover, their scheme is susceptible to “offline password guessing”, “privileged-insider”, and “server secret key leakage” attacks as mentioned in [28].

Esfahani *et al.* [29] designed an authentication protocol for IIoT with low computational cost which is based on the lightweight primitives like one-way hash function and XOR operations. However, their scheme requires to store secret authentication information on an authentication server, which may endanger a single point of failure. Garg *et al.* [30] proposed another lightweight ECC based authentication scheme for IoT based Industry 4.0 application. Though their scheme requires less computation cost, it does not resist against IoT smart device impersonation attack.

Leyou *et al.* [31] introduced a privacy preserving based CP-ABE scheme that supports authority verification without any privacy leakage which provides constant size private keys with short ciphertexts. It is also shown that the selective security under the “Decisional  $n$ -Bilinear Diffie-Hellman Exponent ( $n$ -BDHE)” computational problem with decisional linear assumption, is achieved in this scheme.

Xu *et al.* [32] illustrated a framework for privacy-preserving ABAC system, which assures the security and privacy of the outsourced users’ data stored in “Cloud Service

Provider (CSP)”. The framework also supports secure deduplication which helps to eliminate redundant encrypted data in the CSP with decent communication costs. Tian *et al.* [33] introduced an ABE full privacy protection (ABE-FPP) scheme based on three stages; 1) key generation, 2) access control, and 3) partial decryption. It provides policy hidden strategy, known as hybrid-verification strategy, that reveals only attribute names and also is able to hide its values to preserve privacy during partial decryption.

Later, Gupta *et al.* [34] tried to address the access control problems in the “intelligent transportation system (ITS)” ecosystem by proposing an ABAC system. Their system uses the fine-grained policies with individualized privacy choice in order to grant/deny different activities in the smart entities.

Han *et al.* [35] discussed the “role-based access control (RBAC)” that relies on the analysis using role-permissions matrices and also the implied concept of lattices. They evaluated their methodology by applying it to other substantial practical open-source systems, such as a) MediaWiki, b) Moodle, c) Joomla, and d) WordPress.

Garg *et al.* [36] suggested an authentication protocol for IoT-enabled Industry 4.0 environment that uses “elliptic curve cryptography (ECC)”, “physically unclonable functions (PUFs)” and “hashing function” operations. However, their scheme does not support “session key security under the CK-adversary model”, “dynamic sensor node addition phase” and “blockchain-based solution”.

Amoon *et al.* [37] designed a “role-based reputed access control (RRAC)” scheme for protecting malicious attacks in an IoT system. Their scheme achieves two types of features, where it internally provides an “adaptive certificate based authentication” between users and resources, and it also externally trusts user communication. However, the role of IoT devices is determined separately based on reputation derived by the service provider (SP). In this scheme, precision of reputation is achieved by eliminating untrusted devices that are based on false reputation.

## 2) Blockchain-Based Schemes

Lin *et al.* [38] proposed a blockchain-based secure access control protocol (BSeIn) for industry 4.0 which provides essential security features, such as “authentication”, “auditability”, and “confidentiality”. Moreover, their scheme applies costly bilinear pairing operations that substantially increase the computational overheads. Ren *et al.* [39] designed a “blockchain-based access control scheme for edge based IIoT”. In their scheme, two entities make the session key based on short and long terms secrets, and as a result, their scheme is secure against ESL attack. Since the timestamp is not applied in their scheme, a strong replay attack protection is not provided.

Guangsheng *et al.* [40] introduced a blockchain-based IoT application compatible with the “attribute-based encryption (ABE)”, where the fine-grained access control is used for attributes updation. In addition, they introduced a verification scheme and showed their solution outperforms in searching

complexity and the system revokes the members when there is a direct data leakage. However, Gao *et al.* [41] proposed trustworthy “Ciphertext-Policy Attribute-Based Encryption (CP-ABE)” scheme using ciphertext-policy and attribute hiding access policy with the help of blockchain technology. They used “homomorphic ElGamal cryptosystem” in order to assure the privacy of the attributes.

Zhang *et al.* [42] proposed an “attribute-based access control (ABAC) framework for smart city application” using blockchain smart contract technology. Their scheme consists of a policy management using private Ethereum smart contracts for maintaining policies in ABAC. They computed the cost of gas consumption on Ethereum platform.

Nakamura *et al.* [43] proposed “Capability-Based Access Control (CapBAC)” scheme which stores and manages the capability tokens with local Ethereum-based implementation. However, their scheme fails to resist potential attacks. Moreover, Liu *et al.* [44] presented a CapBAC system using the blockchain technology to regulate the “dynamic identities (DIDs)” for different identities and access rights granting to IoT devices.

In 2023, Yu *et al.* [45] suggested an access control mechanism using the blockchain for an IoT environment. Their approach relies on the Attribute-Based Access Control (ABAC) model. In their scheme, use of the decentralized ABAC model helps for secure decision-making in order to control variables that ensure a fine-grained access control approach.

In 2023, Vangala *et al.* [46] suggested an “efficient blockchain-enabled authenticated key agreement scheme for mobile vehicles-assisted precision agricultural IoT networks, called AgroMobiBlock”. The novelty lies in AgroMobiBlock is that it uses the “elliptic curve operations on an active hybrid blockchain over mobile farming vehicles with low computation and communication costs”.

In 2023, the authors in [47] highlighted various security challenges that are faced for the large IoT based infrastructures like smart cities. Next, they designed a dynamic solution for mitigating the challenges in large IoT based infrastructures with the help of a zero-trust and Attribute-Based Access Control (ABAC) policy, and the blockchain technology.

### 3) Comparative Analysis on Various Features with Existing Approaches

In Table 1, we have provided a comparative analysis of the proposed framework with the state of art existing solutions with respect to various “cryptographic techniques, advantages and limitations” in an IoT environment. The literature study from the existing schemes shows that the existing schemes are either lacking to meet appropriate security requirements or they do not offer blockchain-based solutions in IIoT environment. Moreover, the operational costs due to bilinear pairings make the schemes inefficient and thus, it may not be viable in large scale IIoT system. The recent schemes in “role-based access control (RBAC)”, “attribute-based access control (ABAC)”, and “capability-based access control

(CapBAC)” are mostly centralized cloud-based solutions and they do not provide the appropriate usability of access control with blockchain-based solution in IIoT system. On the other side, the proposed solution in this article offers better security features and more functionality features including protection of session key security under the CK-adversary model [48] and also the blockchain-based solution. Furthermore, the proposed scheme provides “voting-based consensus for verification and addition of a block in blockchain” and resist from possible attacks in blockchain such as transaction privacy leakage, selfish mining attack, balance attack and Sybil attack. Additionally, the proposed scheme is efficient in communication as well as computation as compared to other considered existing competing schemes. Notably, in the proposed scheme (PBACS-PECIIoT), the registration credentials obtained by a smart device ( $SD_i$ ) and the gateway node ( $GN_j$ ) are fetched from the Blockchain during the access control phase for authentication and key agreement purposes. Additionally, the registration credentials stored in the blockchain center ( $BC$ ) are also fetched by an edge server for key management purpose with its gateway node.

## C. RESEARCH CONTRIBUTIONS

The key contributions towards this work are mentioned below.

- We propose a novel “private blockchain-envisioned access control scheme for Pervasive Edge Computing (PEC) in IIoT environment, called PBACS-PECIIoT”. The purpose behind applying the the private blockchain is that the transactions and registration credentials of the entities related to IIoT are confidential and private.
- In the proposed PBACS-PECIIoT, registration credentials obtained by a smart device ( $SD_i$ ) and the gateway node ( $GN_j$ ) are fetched from the Blockchain during the access control phase for authentication and key agreement purposes. Additionally, it is also worth to notice that the registration credentials stored in the blockchain center ( $BC$ ) are fetched by an edge server for key management purpose with its gateway node.
- After collecting the information securely from the deployed IoT smart devices by their respective gateway node(s), the information is securely delivered to the edge servers by their associated gateway nodes in form of transactions. The edge servers are then responsible for building the blocks, verifying and adding them in the private blockchain with the help of the proposed voting-based “Practical Byzantine Fault Tolerance (PBFT)” algorithm [8]. The local ledgers are maintained by the edge servers in the blockchain center.
- A detailed security analysis including the formal security verification has been conducted. It demonstrates that PBACS-PECIIoT is secure against a number of potential attacks against passive/active adversaries.
- The “real testbed experiments for various cryptographic primitives with the help of widely-accepted Multiprecision Integer and Rational Arithmetic Cryptographic Li-

**TABLE 1.** Cryptographic techniques, advantages and limitations of existing authentication/access control schemes in IoT environment

Scheme	Cryptographic Techniques	Advantages	Drawbacks/Limitations	Blockchain Usage
Li <i>et al.</i> [25]	* Elliptic Curve Cryptography (ECC) * Bilinear pairings * Hash functions * Modular exponentiation	* A user is allowed to authorize, revoke, and authenticate for accessing real time information inside IWSN * Signcryption	* Heavy computational overheads due to bilinear pairing operations * Does not support blockchain security solution	—
Bilal and Kang [26]	* Encryption/decryption * Hash function	* Mutual authentication * Key establishment * Establishment of multiple concurrent sessions among sensor nodes	* Vulnerable to parallel-session hijacking attack * Does not support blockchain solution	—
Xue <i>et al.</i> [27]	* Hash functions * Signcryption * Encryption/decryption	* Authentication * Secure information access * Unified storage provision	* Does not provide “key agreement” * Vulnerable to “Ephemeral Secret Leakage (ESL) attack under CK-adversary model	Passive blockchain
Li <i>et al.</i> [30]	* ECC * One-way hash function * Fuzzy extractor for biometric verification	* Mutual authentication * Session key agreement	* Vulnerable to “Ephemeral Secret Leakage (ESL) attack under CK-adversary model * Does not support security blockchain solution	—
Luo <i>et al.</i> [29]	* ECC * One-way hash function * Cross domain heterogeneous signcryption (CDHSC) * Bilinear pairings	* Authentication * Session key agreement	* Does not support dynamic sensor node addition phase * Does not support blockchain solution * Computationally costly due to bilinear pairing operations	—
Lin <i>et al.</i> [42]	* One-way hash function * Hashed message authentication code (HMAC) * Modular exponentiation	* Mutual authentication * Session key agreement * User anonymity * Supports blockchain solution	* Computationally costly	Passive blockchain
Zeng <i>et al.</i> [31]	* ECC * Bilinear pairings	* Online login and authentication * Password change phase	* Insecure against offline password guessing * Vulnerable to privileged-insider attack * Vulnerable to server secret key leakage * Does not support dynamic IoT device addition phase * Does not support blockchain solution	—
Ren <i>et al.</i> [43]	* ECC * One-way hash functions	* Mutual authentication * Session key agreement * Supports blockchain solution	* Vulnerable to replay attack * Does not support dynamic IoT device addition phase	Passive blockchain
Esfahani <i>et al.</i> [33]	* One-way hash function * Bitwise XOR operation	* Mutual authentication * Session key agreement	* Single point of failure * Does not support dynamic IoT device addition phase * Does not support blockchain solution	—
Garg <i>et al.</i> [34]	* ECC * Physically Unclonable Functions (PUFs)	* Mutual authentication * Session key agreement	* Vulnerable to IoT smart device impersonation attack * Does not support dynamic IoT device addition phase * Does not support blockchain solution	—
Li <i>et al.</i> [28]	* ECC * One-way hash function * Fuzzy extractor for biometric verification * Encryption/decryption	* Mutual authentication * Session key agreement	* Does not provide forward security * Insecure against mobile device loss attack * Does not support security blockchain solution	—
Proposed (PBACS-PECIIoT)	* ECC * One-way hash function * Symmetric key encryption	* Mutual authentication * Session key agreement * Supports blockchain solution	* Implementation of real testbed experiments for access control and key management parts	Active hybrid blockchain

brary (MIRACL)” [49] have been performed under both server and Raspberry PI 3 platforms. These testbed experiments measure the computational time for the primitives with respect to these platforms. Moreover, a detailed comparative analysis among PBACS-PECIIoT and other related existing schemes has been performed. It shows the effectiveness and robustness of PBACS-

PECIIoT over other schemes.

- The proposed PBACS-PECIIoT is also implemented through blockchain simulation study in order to measure its performance as well as computational time.

#### D. EVALUATION METRICS

The following evaluation metrics are crucial in the design of an access control scheme in an IIoT environment that used the blockchain technology:

- **Scalability:** The designed access control scheme should support a large number of IoT devices. As a result, it must be also flexible enough against a substantial increase in the network size even after the initial deployment of the IoT devices nodes in the network. This demands for dynamic IoT devices deployment after the initial deployment.
- **Use of active hybrid blockchain:** In an access control scheme, the blockchain technology should not only be used for storing the authenticated data in the blockchain, but also be used during the access control procedure as well to retrieve the registration credentials of the entities from the blockchain as well.
- **Storage cost:** The amount of memory needed for storing the security credentials particularly in the IoT devices (which are typically resource limited) should be kept in minimum.
- **Communication cost:** The number of messages that need to be exchanged during the authentication procedure in the designed access control scheme should be less as possible.
- **Computational cost:** It is measured as the amount of processor cycles needed to authenticate and establish a shared session key between two communicating entities in the IoT network. It needs to be minimum particularly for the resource limited IoT devices.
- **Resilience against physical IoT device capture attack:** Sometimes, the IoT devices can not be monitored always  $24 \times 7$ . Hence, an adversary  $\mathcal{A}$  can physically capture an IoT device and extract all the information stored in its memory using the “power analysis attacks” as mentioned in [50] (see in the threat model discussed in Section II-B). The resilience against IoT device capture attack is measured by “estimating the fraction of total secure communications that are compromised by a capture of  $n_c$  devices *not including* the communication in which the compromised devices are directly involved” as in [51]. Let us say  $P_e(n_c)$  will be the “fraction of total secure communications compromised after capturing  $n_c$  IoT devices by the adversary  $\mathcal{A}$  in the IoT network”. Now, if  $P_e(n_c) = 0$ , an access control scheme will be called as *unconditionally secure against IoT device capture or perfectly resilience against IoT device capture*.
- **Resilience against general attacks:** In an IoT network, the attacks like “replay”, “Man-in-the-Middle (MiTM)”, “impersonation”, and “privileged-insider” attacks should be resisted in an access control scheme like other networks. Moreover, a more important attack, known as “Ephemeral Secret Leakage (ESL)” attack should be protected in the aspect of the session key security.

- **Resilience against blockchain-related attacks:** In an access control scheme that uses the blockchain technology, the blockchain related attacks, such as “transaction privacy leakage”, “selfish mining”, “balance” and “Sybil” attacks should be also protected against an adversary.

#### E. PAPER ORGANIZATION

In the next section, we discuss the system model by considering both the network and threat models. In Section III, the detailed phase-wise discussion of various phases related to PBACS-PECIIoT is provided. The security analysis on PBACS-PECIIoT is provided in sections IV and V. In Section VI, we perform the testbed experiments with the help of widely-accepted “Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL)” [49]. Next, a detailed comparative study among PBACS-PECIIoT and other existing relevant schemes is provided in Section VII. The blockchain implementation is shown on the proposed PBACS-PECIIoT in Section VIII. Finally, the paper is ended with concluding remarks in Section IX.

## II. SYSTEM MODEL

In this section, the network as well as threat models used in the proposed scheme (PBACS-PECIIoT) are discussed.

### A. NETWORK MODEL

The network model used in the proposed PBACS-PECIIoT is shown in Figure 1. The model shows different types of IIoT applications, such as mobile, car, aerospace, and food manufacturing industry.

Various smart IoT devices are attached with each unit of an industry, and all the smart devices, say  $(SD_i | i = 1, 2, 3, \dots, n_{sd})$  are connected with the associated gateway node(s), say  $(GN_j | j = 1, 2, 3, \dots, n_{gn})$ . Each  $GN_j$  is connected with an edge server, say  $(ES_l | l = 1, 2, 3, \dots, n_{es})$ . The registration of all the entities  $(SD_i, GN_j, ES_l)$  is executed by a trusted registration authority, say  $(RA_k | k = 1, 2, 3, \dots, n_{ra})$  for a particular application. Here,  $n_{sd}$ ,  $n_{gn}$ ,  $n_{es}$  and  $n_{ra}$  represent the number of IoT smart devices, gateway nodes, edge servers and RAs, respectively. All the registered  $ES_l$  form a P2P edge servers network, which is also called as the blockchain center. An edge server, being a leader node, say  $ES_l$ , runs a consensus algorithm for creating, verifying, adding, and also mining the blocks in their *local ledgers* of the blockchain center.

### B. THREAT MODEL

The involved entities in an “IIoT environment” need to communicate over insecure channels. Therefore, an adversary  $\mathcal{A}$  can take an opportunity to manipulate/compromise the data exchanged between them. In this paper, we adapt the broadly-accepted “Dolev-Yao threat model (known as DY model)” [52]. Under the DY model,  $\mathcal{A}$  “not only eavesdrops, but can also modify, delete and insert fake information during the communication among the entities”. In addition, we also

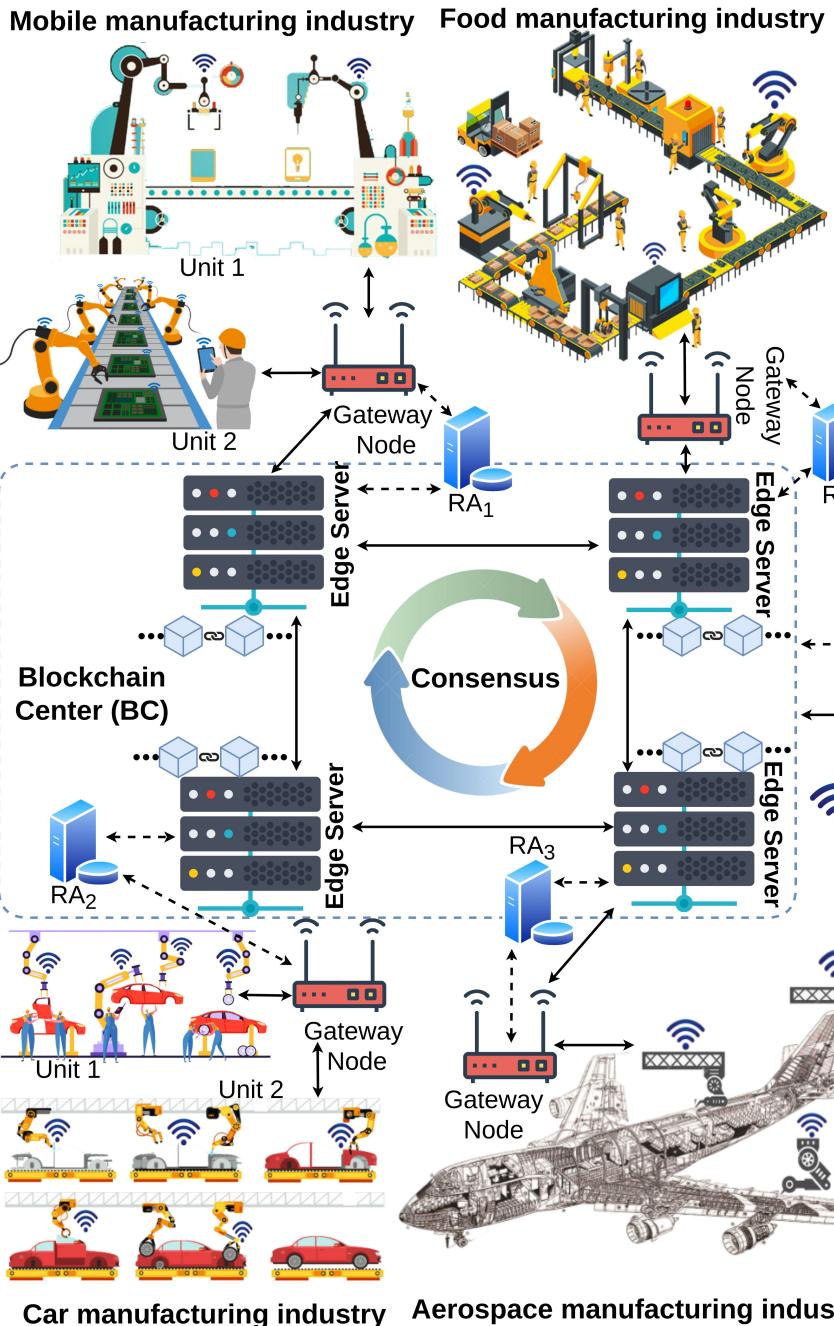


FIGURE 1. Blockchain-envisioned edge-based IIoT environment

adapt the widely-known “Canetti and Krawczyk’s model (CK-adversary model)” [48] which is presently a *de facto* threat model as compared to the DY model. Under the “CK-adversary model”,  $\mathcal{A}$  can compromise the “secret credentials shared between two communicating parties. This results the adversary  $\mathcal{A}$  to compromise the past or future established session keys between the communicating parties by means of compromising the session states and session keys”.

We assume that end-point entities (IoT smart devices) are not trusted, whereas the gateway nodes and edge servers are semi-trusted and the registration authorities are fully

trusted. Since it may not be possible to monitor the IoT smart devices in  $24 \times 7$ , a physical theft of some IoT smart devices by  $\mathcal{A}$  may happen. It may then lead to compromise the secret credentials stored in the captured devices using some sophisticated “power analysis attacks” as mentioned in [50].

### III. THE PROPOSED SCHEME

In this section, different phases relevant to the proposed private blockchain-envisioned access control scheme for edge-based IIoT environment, PBACS-PECIIoT, has been de-

signed.

The proposed PBACS-PECIIoT has various phases, like *registration, access control, key management, and block creation, verification and addition in blockchain*. In Figure 2, we have illustrated the complete process in PBACS-PECIIoT. Note that in this paper, we consider the access control that mainly consists of the following two tasks [53], [54]:

**Task 1 (Node authentication):** This task permits that the newly deployed IoT smart devices and their nearby accessible gateway nodes must authenticate themselves to their neighbor nodes for proving the fact that they are genuinely deployed nodes in the network, and they can also access the network for services.

**Task 2 (Key agreement):** This task is needed for the nodes to establish secret (session) keys with the neighbor nodes (for instance, between an “IoT smart device” and its associated “gateway node”) to assure secure communication after the node authentication process is completed.

The idea behind the design of the proposed scheme is to mutually authenticate two communicating entities through the access control mechanism. It helps in establishing the secret session keys between the authorized entities in the IoT network so that they can secure communicate among each other for secure data delivery. In addition, the key management process between a gateway node and its associate edge server helps in secure communication among them.

Another interesting novelty in the proposed scheme is that as compared to other existing schemes in the literature, the registration credentials obtained by a smart device ( $SD_i$ ) and the gateway node ( $GN_j$ ) are obtained from the Blockchain during the access control phase for authentication and key agreement purposes of the proposed PBACS-PECIIoT. Moreover, the registration credentials stored in the blockchain center ( $BC$ ) are also extracted by an edge server for key management purpose with its gateway node. The existing works on authentication in IoT networks use the passive usage of blockchains with high costs. However, in the present scheme (PBACS-PECIIoT), we have not only utilized the passive blockchain for the secure storage purpose, but also active blockchain during the retrieval of registration credentials during the registration processes. As a result, the active hybrid blockchain has been utilized for strong security in the proposed scheme.

The deployed IoT smart devices first send the messages encrypted with their established session keys during the “access control phase” in Section III-B to their respective gateway node(s). The gateway nodes then send the information encrypted with their secret keys established during the “key management phase” in Section III-C to their respective edge servers. An in-charge edge server is responsible to create a block containing the encrypted transactions of information received from the gateway node(s) or IoT smart device(s) for a particular application.

Since PBACS-PECIIoT makes use of current system timestamps for safeguarding replay attacks, all the communicating entities, like “IoT smart devices”, “gateway nodes”

and “edge servers”, are synchronized with their clocks. It is a widely accepted presumption applied in “different existing authentication and access control approaches under individual networking scenarios” [55], [56]. The list of symbols tabulated in Table 2 are utilized in describing and analyzing the proposed PBACS-PECIIoT. We have provided this notation table for better presentation and understanding of the proposed PBACS-PECIIoT as well. In the proposed scheme, we have used a “non-singular elliptic curve  $E_q(\alpha, \beta) : y^2 = x^3 + \alpha x + \beta \pmod{q}$  over a finite (Galois) field  $GF(q)$ , where  $\alpha, \beta \in Z_q = \{0, 1, 2, \dots, q - 1\}$  are constants with  $4\alpha^3 + 27\beta^2 \neq 0 \pmod{q}$ . In an elliptic curve, there are two types of operations supported: 1) point addition and 2) scalar multiplication. If  $P = (x_P, y_P)$  and  $Q = (x_Q, y_Q)$  be two points in  $E_q(\alpha, \beta)$ , then  $R = (x_R, y_R) = P + Q$  is calculated by the following rule [57]:

$$x_R = (\mu^2 - x_P - x_Q) \pmod{p}$$
$$y_R = (\mu(x_P - x_R) - y_P) \pmod{p}$$
$$\text{where } \mu = \begin{cases} \frac{y_Q - y_P}{x_Q - x_P} \pmod{p}, & \text{if } P \neq -Q \\ \frac{3x_P^2 + a}{2y_P} \pmod{p}, & \text{if } P = Q. \end{cases}$$

The case  $P = Q$  is often referred as doubling the point, and it is represented as  $2.P$ . In particular,  $k \cdot P = P + P + \dots + P$  ( $k$  times), is called elliptic curve point multiplication, where  $k \in Z_q^* = \{1, 2, \dots, q - 1\}$ .

#### A. REGISTRATION PHASE

During registration process of all the communicating entities, like “IoT smart devices”, “gateway nodes” and “edge servers”, each registration authority  $RA_k$  ( $k = 1, 2, \dots, n_{ra}$ ) selects the following system parameters. First of all, each  $RA_k$  will pick a large prime  $q$  and a “non-singular elliptic curve  $E_q(\alpha, \beta) : y^2 = x^3 + \alpha x + \beta \pmod{q}$  over a finite (Galois) field  $GF(q)$  with two constants  $\alpha, \beta \in Z_q$  such that the Elliptic Curve Discrete Logarithm Problem (ECDLP) becomes intractable due to sufficiently chosen large prime  $q$ . For instance, to make ECDLP intractable,  $q$  should be chosen at least 160 bits such that 160-bit “Elliptic Curve Cryptography (ECC)” security remains same as that for an 1024-bit RSA public key cryptosystem [58]. In addition, each  $RA_k$  also selects a base point  $G_k$  corresponding to the chosen elliptic curve  $E_q(\alpha, \beta)$  whose order will be as big as  $q$ , and a common collision resistant cryptographic hash function  $h(\cdot)$  (for example, we can apply the Secure Hash Algorithm (SHA-256) hash function). Furthermore, each  $RA_k$  picks its own secret (private) master key  $mk_{RA_k}$  which is kept secret to itself, computes the respective public key  $Pub_{RA_k} = mk_{RA_k} \cdot G$  and makes the system parameters  $\{E_q(\alpha, \beta), G_k, h(\cdot), Pub_{RA_k}\}$  as public.

##### 1) IoT Smart Devices Registration

This phase occurs in offline mode prior to deployment of the IoT smart devices in their respective deployment areas by the associated registration authority  $RA_k$ . Note that the private and public key pairs for the IoT smart devices are generated

**TABLE 2.** Notations and their meanings

Symbol	Meaning
$RA$	Trusted registration authority
$SD_i, RID_{SD_i}$	$i^{th}$ IoT smart device and its pseudo identity
$GN_j, RID_{GN_j}$	$j^{th}$ gateway node and its pseudo identity
$ES_l, RID_{ES_l}$	$l^{th}$ edge server and its pseudo identity
$n_{sd}, n_{ra}, n_{gn}, n_{es}$	Number of IoT smart devices, RAs, gateway nodes and edge servers, respectively
$h(\cdot)$	A collision-resistant cryptographic one-way hash function
$q$	A large prime number
$GF(q)$	Galois finite field over prime $q$
$E_q(\alpha, \beta)$	A non-singular elliptic curve: $y^2 = x^3 + \alpha x + \beta \pmod{q}$ over $GF(q)$ with $\alpha, \beta \in Z_q = \{0, 1, \dots, q-1\}$
$G_k$	A base point in $E_q(\alpha, \beta)$
$P + Q$	Elliptic curve point addition of two points $P, Q \in E_q(\alpha, \beta)$ ,
$u \cdot G_k$	Elliptic curve point multiplication;
$u \cdot v$	$u \cdot G_k = G_k + G_k + \dots + G_k$ ( $u$ times), $G_k \in E_q(\alpha, \beta)$ , $u \in Z_q^*$
$(pr_X, Pub_X)$	Modular multiplication of elements $u, v \in Z_q$
$E(\cdot)/D(\cdot)$	Signature private-public key pair of an entity $X$ , where $Pub_X = pr_X \cdot G_k$
$TS_x$	ECC encryption/decryption algorithm
$\Delta T$	Current system timestamp generated by an entity $X$
$\parallel, \oplus$	Maximum transmission delay associated with a message
	Data concatenation and exclusive-OR operators, respectively

by the registration authority  $RA_k$ , and these are pre-installed in IoT devices' memory before placing them in the network.

For registering each deployed IoT smart device  $SD_i$  for a particular application of IIoT, the respective registration authority  $RA_k$  first selects a unique identity  $ID_{SD_i}$  and then computes the corresponding pseudo-identity  $RID_{SD_i} = h(ID_{SD_i} || mk_{RA_k})$  and temporal credential  $TC_{SD_i} = h(RID_{SD_i} || mk_{RA_k} || RTS_{SD_i} || pr_{SD_i})$  of  $SD_i$ , where the private key of each  $SD_i$  is a random secret  $pr_{SD_i} \in Z_q^* = \{1, 2, \dots, q-1\}$  and its public key is  $Pub_{SD_i} = pr_{SD_i} \cdot G_k$ , and  $RTS_{SD_i}$  is the registration timestamp of  $SD_i$ . The  $RA_k$  stores the information  $\{RID_{SD_i}, pr_{SD_i}, Pub_{RA_k}\}$  into its secure memory prior to placement in IIoT application, and makes  $Pub_{SD_i}$  as public.

After that, the  $RA$  sends the registration related credentials  $RegCred_{SD_i} = \{RID_{SD_i}, TC_{SD_i}, Pub_{SD_i}, E_q(\alpha, \beta), G_k\}$  to the blockchain center ( $BC$ ) in the form of a transaction, say  $Tx_{RegCred_{SD_i}} = \langle RID_{SD_i}, E_{Pub_{SD_i}}[RegCred_{SD_i}], ECDSA.Sign_{mk_{RA_k}}[RegCred_{SD_i}] \rangle$ , where  $E_{Pub_X}(\cdot)$  and  $D_{pr_X}(\cdot)$  represent the "ECC-based encryption and decryption using the public key  $Pub_X$  and private key  $pr_X$  of an entity," respectively, and  $ECDSA.Sign(\cdot)$  and  $ECDSA.Ver(\cdot)$  denote the "elliptic curve digital signature algorithm (ECDSA)-based signature generation and verification methods," respectively.

$RA_k$  deletes  $RID_{SD_i}$ ,  $TC_{SD_i}$  and  $(pr_{SD_i}, Pub_{SD_i})$  from its database for "security reasons in order to avoid privileged-insider attack". It is worth noticing that the registration credentials need not to be sent back to the IoT devices after the registration process, because the credentials are directly stored in the memory of the deployed smart devices

as well as in the blockchain center prior to their placement in the network.

**Remark 1.** During the IoT smart devices registration process, an IoT device  $SD_i$  needs to store the registration credentials  $\{RID_{SD_i}, pr_{SD_i}, Pub_{RA_k}\}$  in its memory. If we assume that a "random number", an "identity", a "one-way hash function (using SHA-256 hashing algorithm)", an "elliptic curve point  $P \in E_q(\alpha, \beta)$ " and a "timestamp" are 160, 160, 256, 320, and 32 bits, respectively, the storage cost for  $SD_i$  becomes  $(256 + 160 + 320) = 736$  bits only. As a result, the proposed scheme is efficient in storage overhead particularly for the IoT smart devices, which are resource limited as compared to other nodes like gateway nodes and edge servers.

## 2) Gateway Nodes Registration

To register a gateway node  $GN_j$  belonging to a particular IIoT application, its respective  $RA_k$  first picks the unique identity  $ID_{GN_j}$  and then computes its pseudo-identity  $RID_{GN_j} = h(ID_{GN_j} || mk_{RA_k})$  and temporal credential  $TC_{GN_j} = h(ID_{GN_j} || RTS_{GN_j} || mk_{RA_k})$  for  $GN_j$ , where  $RTS_{GN_j}$  is  $GN_j$ 's registration time. Next,  $RA_k$  picks a " $t$ -degree symmetric bivariate polynomial over the finite (Galois) field  $GF(q)$  as  $g_j(x, y) = \sum_{u=0}^t \sum_{v=0}^t a_{u,v} x^u y^v$ , where the co-efficients  $a_{u,v}$  are from  $Z_q$ ,  $g_j(x, y) = g_j(y, x)$ , and  $t >> n_{gn}$  and  $t >> n_{es}$ ", as in Blundo *et al.*'s scheme [59]. Furthermore,  $RA_k$  calculates the polynomial share  $g_j(RID_{GN_j}, y) = \sum_{u=0}^t \sum_{v=0}^t a_{u,v} (RID_{GN_j})^u y^v$  over  $GF(q)$  and sends the registration credential  $\{RID_{GN_j}\}$  to  $GN_j$  via a secure channel (for example, in person).

After receiving the registration credentials from

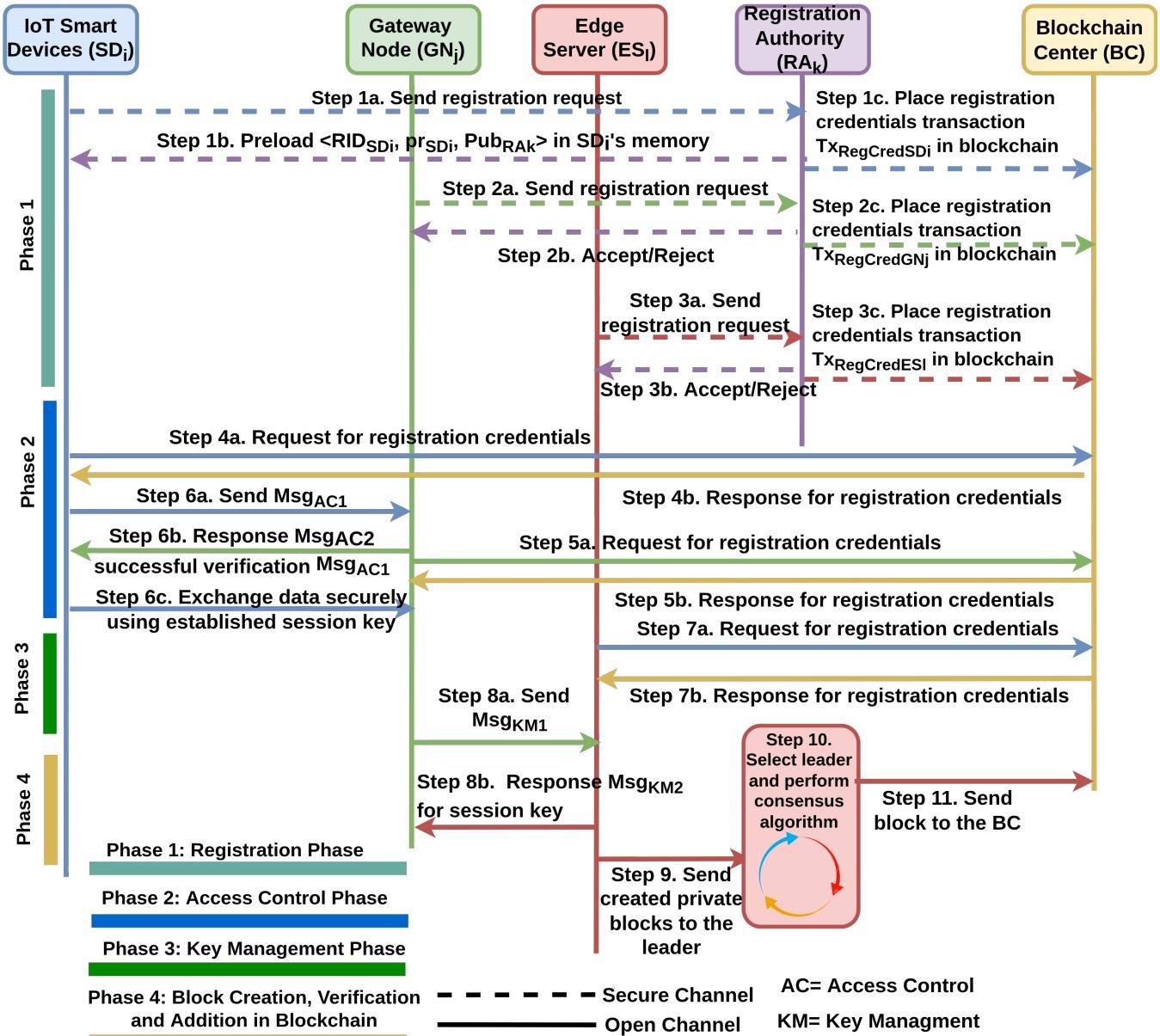


FIGURE 2. Illustration of complete process in PBACS-PECIoT

$RA_k$ ,  $GN_j$  picks its own random secret (private) key  $pr_{GN_j} \in Z_q^*$ , computes the respective public key  $Pub_{GN_j} = pr_{GN_j} \cdot G_k$ , stores the private key  $pr_{GN_j}$  in its secure database and publishes the public key  $Pub_{GN_j}$ . Next, the  $RA_k$  sends the registration credentials  $RegCred_{GN_j} = \{RID_{GN_j}, TC_{GN_j}, g_j(RID_{GN_j}, y), E_q(\alpha, \beta), G_k, \{(RID_{SD_i}, Pub_{SD_i})\}, Pub_{GN_j}\}$  to the blockchain center ( $BC$ ) in the form of a transaction, say  $Tx_{RegCred_{GN_j}} = \langle RID_{GN_j}, E_{Pub_{GN_j}}[RegCred_{GN_j}], ECDSA.Sign_{mk_{RA_k}}[RegCred_{GN_j}] \rangle$ . Note that only for the IoT smart devices  $SD_i$  that are associated with  $GN_j$  in a particular IIoT application,  $\{(RID_{SD_i}, Pub_{SD_i})\}$  are available to  $GN_j$ .  $RA_k$  also deletes  $RID_{GN_j}$  and  $TC_{GN_j}$  from its database for security reasons in order to

avoid “privileged-insider attack”, and stores  $Pub_{GN_j}$  in each associated  $SD_i$ 's memory. Finally,  $GN_j$  stores the credentials  $\{RID_{GN_j}, pr_{GN_j}\}$  in its secure database.

### 3) Edge Servers Registration

To register an edge server  $ES_l$  belonging to one or more  $GN_j$  for a particular IIoT application, the in-charge  $RA_k$  picks a unique identity  $ID_{ES_l}$  for  $ES_l$  and computes the pseudo-identity  $RID_{ES_l} = h(mk_{RA_k} || ID_{ES_l})$  and the polynomial share  $g_j(RID_{ES_l}, y) = \sum_{u=0}^t \sum_{v=0}^t a_{u,v}(RID_{ES_l})^u y^v$  over  $GF(q)$ . It is worth noticing that  $g_j(x, y)$  is the common polynomial shared between  $GN_j$  and  $ES_l$ . After that  $RA_k$  sends the registration credentials  $\{RID_{ES_l}\}$  to  $ES_l$  via secure channel.

Once the registration credentials are received by  $ES_l$  from  $RA_k$ ,  $ES_l$  picks its own random private key  $pr_{ES_l} \in Z_q^*$ , calculates the corresponding public key  $Pub_{ES_l} = pr_{ES_l} \cdot G_k$ , and publishes  $Pub_{ES_l}$  as public. Here, the polynomial  $g_j(x, y)$  is used to setup a symmetric secret key between the “gateway node  $GN_j$ ” and its associated “edge server  $ES_l$ ”, which is further utilized in establishing the session key  $SK_{ES_l, GN_j}$  ( $SK_{GN_j, ES_l}$ ) between them for secret communications (see Section III-C).

The  $RA$  sends the registration credentials  $RegCred_{ES_l} = \{RID_{ES_l}, g_j(RID_{ES_l}, y), TC_{ES_l}, Pub_{ES_l}, E_q(\alpha, \beta), G_k\}$  to the blockchain center ( $BC$ ) in the form of a transaction, say  $Tx_{RegCred_{ES_l}} = \langle RID_{ES_l}, E_{Pub_{ES_l}}[RegCred_{ES_l}], ECDSA.Sign_{mk_{RA_k}}[RegCred_{ES_l}] \rangle$ . Finally,  $ES_l$  needs to store the credentials  $\{RID_{ES_l}, pr_{ES_l}\}$  in its secure database.

**Remark 2.** It is noted that the IoT smart devices registration phase occurs in offline mode prior to deployment of the IoT devices in their respective deployment areas by the associated  $RA_k$ . In addition, the registration of the gateway nodes  $GN_j$  and the edge servers  $ES_l$  are executed in secure channels by the  $RA_k$ . As a result, there is no possibility of the impersonation attacks by an adversary (including the insider attacker) at the device/gateway/edge server registration phases because the encrypted registration credentials along with their signatures are placed into the blockchain.

## B. ACCESS CONTROL PHASE

It is done between a “registered IoT smart device ( $SD_i$ )” and its respective “gateway node ( $GN_j$ )” for a particular application in IIoT. This phase helps to perform a “mutual authentication and session key establishment between  $SD_i$  and  $GN_j$ ”. Before initiating the access control process, the  $SD_i$  and  $GN_j$  need to obtain the registration credentials that are already stored in the blockchain center ( $BC$ ). Note that this is executed only once because the registration credentials obtained from the  $BC$  can be stored in the secure databases of both  $SD_i$  and  $GN_j$ . For this purpose, the following steps are involved:

- *Registration credentials obtained by smart device ( $SD_i$ ):*  $SD_i$  first sends a request message  $RegCredReq_{SD_i} = \{RID_{SD_i}\}$  to the  $BC$  over open channel for obtaining its registration credentials. After receiving the request, the  $BC$  checks  $RID_{SD_i}$  and fetches the transaction  $Tx_{RegCred_{SD_i}} = \langle RID_{SD_i}, E_{Pub_{SD_i}}[RegCred_{SD_i}], ECDSA.Sign_{mk_{RA_k}}[RegCred_{SD_i}] \rangle$  and sends it to  $SD_i$  over public channel.  $SD_i$  upon receiving  $Tx_{RegCred_{SD_i}}$ , decrypts  $E_{Pub_{SD_i}}[RegCred_{SD_i}]$  using the public key  $Pub_{RA_k}$  of the associated  $RA_k$  to extract  $RegCred_{SD_i} = \{RID_{SD_i}, TC_{SD_i}, Pub_{SD_i}, E_q(\alpha, \beta), G_k\}$ . Now, if the decrypted  $RID_{SD_i}$  matches with its received version,  $SD_i$  further validates the signature  $ECDSA.Sign_{mk_{RA_k}}[RegCred_{SD_i}]$  by applying the  $ECDSA.Ver(\cdot)$  algorithm using the  $RA_k$ 's public key  $Pub_{RA_k}$ . If the signature is valid,  $SD_i$  then only stores

$RegCred_{SD_i}$  in its memory for the mutual authentication and key establishment purpose.

- *Registration credentials obtained by gateway node ( $GN_j$ ):*  $GN_j$  also sends a request message  $RegCredReq_{GN_j} = \{RID_{GN_j}\}$  for obtaining the registration credentials to the  $BC$  over open channel. Once the request is processed by the  $BC$ , the  $BC$  checks  $RID_{GN_j}$ , fetches the transaction  $Tx_{RegCred_{GN_j}} = \langle RID_{GN_j}, E_{Pub_{GN_j}}[RegCred_{GN_j}], ECDSA.Sign_{mk_{RA_k}}[RegCred_{GN_j}] \rangle$  corresponding to  $RID_{GN_j}$  and sends it to  $GN_j$  over public channel. Moreover,  $GN_j$  upon receiving  $Tx_{RegCred_{GN_j}}$ , decrypts  $E_{Pub_{GN_j}}[RegCred_{GN_j}]$  using the public key  $Pub_{RA_k}$  of the associated  $RA_k$  to extract  $RegCred_{SD_i} = \{RID_{SD_i}, TC_{SD_i}, Pub_{SD_i}, E_q(\alpha, \beta), G_k\}$ . On successful matching of the decrypted  $RID_{GN_j}$  with its received version,  $GN_j$  checks the validity of the signature  $ECDSA.Sign_{mk_{RA_k}}[RegCred_{GN_j}]$  using the public key  $Pub_{RA_k}$ . Upon successful signature validation,  $GN_j$  stores  $RegCred_{GN_j}$  in its secure database for the mutual authentication and key establishment purpose.

We now discuss the following steps needed for the access control between  $SD_i$  and  $GN_j$  with the help of the obtained registration credentials  $RegCred_{SD_i}$  and  $RegCred_{GN_j}$  from the  $BC$ , respectively.

- *Step AC1.*  $SD_i$  picks a random secret  $rs_{SD_i} \in Z_q^*$  and the current timestamp  $TS_{SD_i}$  for computing  $RS_{SD_i} = h(TC_{SD_i} || rs_{SD_i} || pr_{SD_i} || RID_{SD_i} || TS_{SD_i}) \cdot G_k$ . Furthermore,  $SD_i$  computes signature on  $rs_{SD_i}$  as  $Sig_{SD_i} = h(TC_{SD_i} || rs_{SD_i} || pr_{SD_i} || RID_{SD_i} || TS_{SD_i}) + h(RID_{SD_i} || Pub_{SD_i} || Pub_{GN_j} || TS_{SD_i}) * pr_{SD_i} \pmod{q}$ .  $SD_i$  then sends access control request message  $Msg_{AC1} = \{RID_{SD_i}, Sig_{SD_i}, RS_{SD_i}, TS_{SD_i}\}$  to its corresponding gateway node  $GN_j$  via open channel.
- *Step AC2.* If the message  $Msg_{AC1}$  is received at time  $TS_{SD_i}^*$ , the  $GN_j$  first checks timeliness of received  $TS_{SD_i}$  by verifying  $|TS_{SD_i}^* - TS_{SD_i}| < \Delta T$ , where  $\Delta T$  signifies the “maximum transmission delay with a message”. If it is valid,  $GN_j$  retrieves  $Pub_{SD_i}$  corresponding to  $RID_{SD_i}$  from its own database and verifies the received signature  $Sig_{SD_i}$  by the condition:  $Sig_{SD_i} \cdot G_k = RS_{SD_i} + h(RID_{SD_i} || Pub_{SD_i} || Pub_{GN_j} || TS_{SD_i}) \cdot Pub_{SD_i}$ . Upon successful signature validation,  $GN_j$  validates  $SD_i$  as authentic device, creates a random secret  $rs_{GN_j} \in Z_q^*$  and the current timestamp  $TS_{GN_j}$  for calculating  $RS_{GN_j} = h(TC_{GN_j} || rs_{GN_j} || pr_{GN_j} || RID_{GN_j} || TS_{GN_j}) \cdot G_k$  and the Diffie-Hellman type key  $DHK_{GN_j, SD_i} = h(TC_{GN_j} || rs_{GN_j} || pr_{GN_j} || RID_{GN_j} || TS_{GN_j}) \cdot RS_{SD_i}$ . Furthermore,  $GN_j$  evaluates its own polynomial share

$g_j(RID_{GN_j}, y)$  at the point  $y = RID_{SD_i}$  to obtain  $g_j(RID_{GN_j}, RID_{SD_i})$ ,  
 $y_{GN_j} = h(g_j(RID_{GN_j}, RID_{SD_i}) \parallel Sig_{SD_i} \parallel TS_{GN_j}) \oplus h(DHK_{GN_j, SD_i} \parallel TS_{SD_i} \parallel TS_{GN_j})$ ,  
and also computes the signature on  $rs_{GN_j}$  and  $DHK_{GN_j, SD_i}$  as  
 $Sig_{GN_j} = h(TC_{GN_j} \parallel rs_{GN_j} \parallel pr_{GN_j} \parallel RID_{GN_j} \parallel TS_{GN_j}) + h(RID_{GN_j} \parallel RID_{SD_i} \parallel Pub_{GN_j} \parallel DHK_{GN_j, SD_i} \parallel y_{GN_j}) * pr_{GN_j} \pmod{q}$ .

Next,  $GN_j$  dispatches the access control response message  $Msg_{AC2} = \{RID_{GN_j}, Sig_{GN_j}, RS_{GN_j}, y_{GN_j}, TS_{GN_j}\}$  to its corresponding  $SD_i$  via open channel.

- Step AC3. Let  $SD_i$  receive the message  $Msg_{AC2}$  at time  $TS_{GN_j}^*$ .  $SD_i$  then checks  $TS_{GN_j}$ 's validity by  $|TS_{GN_j}^* - TS_{GN_j}| < \Delta T$  and if it is valid,  $SD_i$  fetches  $Pub_{GN_j}$  corresponding to  $RID_{GN_j}$  from its memory.  $SD_i$  calculates the Diffie-Hellman type key

$DHK_{SD_i, GN_j} = h(TC_{SD_i} \parallel rs_{SD_i} \parallel pr_{SD_i} \parallel RID_{SD_i} \parallel TS_{SD_i}) \cdot RS_{GN_j}$ ,  
 $z_{SD_i} = y_{GN_j} \oplus h(DHK_{SD_i, GN_j} \parallel TS_{SD_i} \parallel TS_{GN_j})$ , which should be equal to  $h(g_j(RID_{GN_j}, RID_{SD_i}) \parallel Sig_{SD_i} \parallel TS_{GN_j})$ , and verifies the signature by the condition:

$$Sig_{GN_j} \cdot G_k = RS_{GN_j} + h(RID_{GN_j} \parallel RID_{SD_i} \parallel Pub_{GN_j} \parallel DHK_{SD_i, GN_j} \parallel y_{GN_j}) \cdot Pub_{GN_j}.$$

Upon successful signature validation,  $SD_i$  authenticates  $GN_j$  as valid, generates current timestamp  $TS'_{SD_i}$ , and computes the session key shared with  $GN_j$  as  $SK_{SD_i, GN_j} = h(DHK_{SD_i, GN_j} \parallel z_{SD_i})$  and its verifier  $SKV_{SD_i, GN_j} = h(SK_{SD_i, GN_j} \parallel TS'_{SD_i})$ .

Finally,  $SD_i$  sends the acknowledgment message  $Msg_{AC3} = \{SKV_{SD_i, GN_j}, TS'_{SD_i}\}$  to  $GN_j$  via public channel.

- Step AC4. Upon reception of the message  $Msg_{AC3}$  at time  $TS''_{SD_i}$ ,  $GN_j$  checks timeliness of received  $TS'_{SD_i}$  by verifying  $|TS''_{SD_i} - TS'_{SD_i}| < \Delta T$ . If the validation passes,  $GN_j$  then calculates the session key shared with  $SD_i$  as

$$SK_{GN_j, SD_i} = h(DHK_{GN_j, SD_i} \parallel h(g_j(RID_{GN_j}, RID_{SD_i}) \parallel Sig_{SD_i} \parallel TS_{GN_j}))$$

and its verifier

$$SKV_{GN_j, SD_i} = h(SK_{GN_j, SD_i} \parallel TS'_{SD_i}).$$

If the verification condition:

$SKV_{GN_j, SD_i} = SKV_{SD_i, GN_j}$  holds good, both  $GN_j$  and  $SD_i$  store the same session key  $SK_{GN_j, SD_i}$  ( $= SK_{SD_i, GN_j}$ ) for their secret communications.

### C. KEY MANAGEMENT PHASE

Before the key management process starts, an edge server ( $ES_l$ ) needs to obtain its registration credentials from the  $BS$  like  $SD_i$  and  $GN_j$  as discussed in Section III-B. Note that  $GN_j$  already obtained its registration credentials from the  $BC$  and stored in its secure database.  $ES_l$  issues a request  $RegCredReq_{ES_l} = \{RID_{ES_l}\}$  for obtaining its registration credentials to the  $BC$

over open channel. Once the  $BC$  checks the validity of  $RID_{ES_l}$ , it fetches the corresponding transaction  $Tx_{RegCred_{ES_l}} = \langle RID_{ES_l}, E_{Pub_{ES_l}}[RegCred_{ES_l}], ECDSA.Sign_{mk_{RA_k}}[RegCred_{ES_l}] \rangle$  and sends it to  $ES_l$  over public channel. After decrypting  $E_{Pub_{ES_l}}[RegCred_{ES_l}]$  using the private key  $pr_{ES_l}$ ,  $ES_l$  extracts  $RegCred_{ES_l} = \{RID_{ES_l}, g_j(RID_{ES_l}, y), TC_{ES_l}, Pub_{ES_l}, E_q(\alpha, \beta), G_k\}$  and checks the validity of both the decrypted  $RID_{ES_l}$  and the signature  $ECDSA.Sign_{mk_{RA_k}}[RegCred_{ES_l}]$ . If all these are valid,  $ES_l$  stores  $RegCred_{ES_l}$  in its secure database which is used for the key management purpose as discussed below.

The sole goal of this phase is to setup a (pairwise) secret key between a gateway node ( $GN_j$ ) and its corresponding edge server ( $ES_l$ ) for their communications. This phase involves the exchange of three messages, namely  $Msg_{KM1}$ ,  $Msg_{KM2}$  and  $Msg_{KM3}$  between  $GN_j$  and  $ES_l$ , that use the registration credentials obtained from the  $BC$  along with random generated secrets and current timestamps. After verifying the message  $Msg_{KM1}$ ,  $ES_l$  generates the session key shared with  $GN_j$  and sends the message  $Msg_{KM2}$  to  $GN_j$ . Validation of  $Msg_{KM2}$  by  $GN_j$  assures mutual authentication between  $GN_j$  and  $ES_l$ . Furthermore, verification of  $Msg_{KM3}$  guarantees that the “session key established between  $GN_j$  and  $ES_l$  are same and legitimate”.

We now explain the followings stages:

- Step KM1. The initiator  $GN_j$  first creates a random secret  $rs_{GN_j} \in Z_q^*$  and a current timestamp  $TS_{GN_j}$  in order to calculate

$$RS_{GN_j} = h(RID_{GN_j} \parallel rs_{GN_j} \parallel TC_{GN_j} \parallel TS_{GN_j} \parallel pr_{GN_j}) \cdot G_k.$$

Next,  $GN_j$  calculates the signature on  $rs_{GN_j}$  as

$$Sig_{GN_j} = h(RID_{GN_j} \parallel rs_{GN_j} \parallel TC_{GN_j} \parallel TS_{GN_j} \parallel pr_{GN_j}) + h(RID_{GN_j} \parallel Pub_{GN_j} \parallel Pub_{ES_l} \parallel TS_{GN_j}) * pr_{GN_j} \pmod{q}$$

and dispatches the request message  $Msg_{KM1} = \{RID_{GN_j}, RS_{GN_j}, Sig_{GN_j}, TS_{GN_j}\}$  to its respective  $ES_l$  via open channel.

- Step KM2. After receiving  $Msg_{KM1}$ , if the timeliness check of the received timestamp  $TS_{GN_j}$  passes,  $ES_l$  proceeds to verify signature  $Sig_{GN_j}$  by the condition:

$$Sig_{GN_j} \cdot G_k = RS_{GN_j} + h(RID_{GN_j} \parallel Pub_{GN_j} \parallel Pub_{ES_l} \parallel TS_{GN_j}) \cdot Pub_{GN_j}.$$

Now, if the signature is valid,  $ES_l$  treats  $GN_j$  as valid, and creates a random secret  $rs_{ES_l} \in Z_q^*$  and current timestamp  $TS_{ES_l}$  to calculate

$$RS_{ES_l} = h(RID_{ES_l} \parallel pr_{ES_l} \parallel rs_{ES_l} \parallel TS_{ES_l}) \cdot G_k$$

and the Diffie-Hellman type key

$$DHK_{ES_l, GN_j} = h(RID_{ES_l} \parallel pr_{ES_l} \parallel rs_{ES_l} \parallel TS_{ES_l}) \cdot RS_{GN_j}.$$

After these computations,  $ES_l$  also evaluates its own polynomial share  $g_j(RID_{ES_l}, y)$  at the point  $y = RID_{GN_j}$  to have the secret  $g_j(RID_{ES_l}, RID_{GN_j})$ , and then computes the secret pairwise key shared with  $GN_j$  as  $SK_{ES_l, GN_j} = h(DHK_{ES_l, GN_j} \parallel g_j(RID_{ES_l}, RID_{GN_j}))$  and a signature on both  $rs_{ES_l}$  and

$SK_{ES_l,GN_j}$  as

$$\begin{aligned} Sig_{ES_{l2}} &= h(RID_{ES_l} \parallel pr_{ES_l} \parallel rs_{ES_{l2}} \parallel TS_{ES_{l2}}) \\ &+ h(Pub_{ES_l} \parallel SK_{ES_l,GN_j} \parallel TS_{ES_{l2}}) * pr_{ES_l} \pmod{q}. \end{aligned}$$

Next,  $ES_l$  dispatches the response message  $Msg_{KM2} = \{RID_{ES_l}, RS_{ES_{l2}}, Sig_{ES_{l2}}, TS_{ES_{l2}}\}$  to  $GN_j$  via open channel.

- **Step KM3.** After checking the timeliness of the timestamp  $TS_{ES_{l2}}$  in the received message  $Msg_{KM2}$ ,  $GN_j$  computes the “Diffie-Hellman type key”

$$\begin{aligned} DHK_{GN_j,ES_l} &= h(RID_{GN_j} \parallel rs_{GN_{j1}} \parallel TC_{GN_j} \parallel TS_{GN_{j1}} \parallel pr_{GN_j}) \cdot RS_{ES_{l2}}, \\ g_j(RID_{GN_j}, RID_{ES_l}) &= g_j(RID_{ES_l}, RID_{GN_j}) \end{aligned}$$

using its own polynomial share  $g_j(RID_{GN_j}, y)$  as  $g_j(x, y) = g_j(y, x)$  and the session key shared with  $ES_l$  as  $SK_{GN_j,ES_l} = h(DHK_{GN_j,ES_l} \parallel g_j(RID_{GN_j}, RID_{ES_l}))$ . Next,  $GN_j$  verifies the signature  $Sig_{ES_{l2}}$  as

$$Sig_{ES_{l2}} \cdot G_k = RS_{ES_{l2}} + h(Pub_{ES_l} \parallel SK_{GN_j,ES_l} \parallel TS_{ES_{l2}}) \cdot Pub_{ES_l}.$$

If the signature validation passes,  $GN_j$  also treats  $ES_l$  as authentic entity.

Finally, both  $GN_j$  and  $ES_l$  require to store the same secret pairwise key  $SK_{GN_j,ES_l}$  ( $= SK_{ES_l,GN_j}$ ) for their secure communications. Both the access control and key management phases are explained briefly in Figure 3.

#### D. BLOCK CREATION, VERIFICATION AND ADDITION IN BLOCKCHAIN

In this section, we elaborate the process of block creation, verification and addition of that block in the blockchain. For this issue, the IoT smart devices first send the messages encrypted with their established session keys as described during the “access control phase” in Section III-B to their respective gateway node(s). In turn, the gateway nodes also send the information encrypted with their secret keys established during the “key management phase” in Section III-C to their respective edge servers. An edge server  $ES_l$  is then responsible to construct a block containing the encrypted transactions of information received from the gateway node(s) or IoT smart device(s) for a particular application. Here, the ECC public key  $Pub_{ES_l}$  is used for generating the encrypted transactions because the information is strictly private and confidential with respect to an IIoT application.  $ES_l$  creates Merkle tree root on the encrypted transactions along with timestamp and random number. The current hash block is computed as  $CurBH = h(BlkV \parallel PreBH \parallel MrkTR \parallel IdtT_m \parallel TS_{ES_l} \parallel CreBID \parallel Pub_{ES_l} \parallel \{E_{Pub_{ES_l}}(Tx_i) \mid i = 1, 2, \dots, t_n\})$  and the signature on  $CurBH$  as  $Sig_{Block_k} = ECDSA.Sig_{pr_{ES_l}}(CurBH)$  where  $ECDSA.Sig(\cdot)$  denotes the “ECDSA signature generation algorithm”. The overall structure of a block  $Block_k$  is shown in Figure 4. The encryption is used in the transactions to make the transactions private with the edge server so that other P2P servers can not decrypt without private key of the particular edge server. Since the encryption is performed with the help of  $Pub_{ES_l}$ , so only particular edge server associated

with an application can see and decrypt the data. Finally, through the consensus algorithm provided in Figure 5, a leader among the group edge servers in the P2P network is selected using the existing leader selection algorithm [60] and then the leader sends the created block, say  $Block_k$  to its peer nodes to have the consensus among them for verifying and adding the block in their local ledgers of blockchain center containing the fog servers. Note that we have applied the “Practical Byzantine Fault Tolerance (PBFT)” algorithm [8] for consensus purpose. However, we have provided the voting-based PBFT version as the proposed consensus algorithm, in which a leader  $L$  selected among the P2P network, generates a current timestamp  $TS_L$  and a random number  $r_L$  to perform voting process.  $L$  then creates signature  $Sig_L$  using the ECDSA signature generation algorithm with its own private key  $pr_L$  on the message  $h(Block_k \parallel TS_L \parallel r_L \parallel VTR_{req})$ , where  $VTR_{req}$  is voting request, and sends the request message  $\langle Block_k, Sig_L, E_{Pub_{ES_l}}[r_L, VTR_{req}], TS_L \rangle$  to each other edge server  $ES_l$  via public channel. After successful validation of timestamp  $TS_L$ , Merkle tree root  $MrkTR$ , current block hash  $CurBH$  and signature on block  $Sig_{Block_k}$ , each other node in the P2P network sends the response message  $\langle E_{Pub_L}[r_L, TS_{ES_l}, VTR_{res}], TS_{ES_l} \rangle$  to  $L$  via public channel, where  $VTR_{res}$  is the “voting response” and  $TS_{ES_l}$  is the “current timestamp”.

An edge server associated with an IIoT application is responsible to create the blocks and store them into the blockchain after the consensus process as described in Figure 5.

**Remark 3.** In this work, we have mainly considered the private blockchain scenario where the data is private and confidential with respect to each edge server. However, there are some applications, where the data needs to be shared inside the system. Thus, encrypting them will make the entities without the secret keys unable to decrypt the data and use the data. In this case, the edge servers can maintain a group (secret) key among them so that the selected shared data (transactions) can be now encrypted with the help of the group key using symmetric key encryption. Hence, the shared encrypted data can be decrypted by other edge servers using the same group key.

#### E. DYNAMIC IOT SMART DEVICE ADDITION PHASE

Due to hostile environment/power exhaustion of IoT smart devices, the devices may be either physically captured or shut down. To continue the functionality of IIoT environment, new smart device, say  $SD^{new}$  needs to be added. Prior to deployment,  $SD^{new}$  is required to register by the trusted registration authority  $RA_k$  in that particular application where existing other smart devices are already there. For registering  $SD^{new}$ ,  $RA_k$  needs to follow the same steps as described in the IoT smart devices registration (see Section III-A1).

Access Control Phase	
IoT smart device ( $SD_i$ )	Gateway node ( $GN_j$ )
Obtain registration credentials from the blockchain center ( $BC$ ). Generates random secret $rs_{SD_i} \in Z_q^*$ , current timestamp $TS_{SD_i}$ . Calculate $RS_{SD_i} = h(TC_{SD_i}    rs_{SD_i}    pr_{SD_i}    RID_{SD_i}    TS_{SD_i}) \cdot G_k$ , $Sig_{SD_i} = h(TC_{SD_i}    rs_{SD_i}    RID_{SD_i}    TS_{SD_i}) + h(RID_{SD_i}    Pub_{SD_i}    Pub_{GN_j}    TS_{SD_i}) * pr_{SD_i} \pmod{q}$ . $Msg_{AC1} = \{RID_{SD_i}, Sig_{SD_i}, RS_{SD_i}, TS_{SD_i}\}$ (via open channel)	Obtain registration credentials from the blockchain center ( $BC$ ). Check if $ TS_{SD_i}^* - TS_{SD_i}  < \Delta T$ ? If valid, retrieve $Pub_{SD_i}$ corresponding to $RID_{SD_i}$ . Verify signature $Sig_{SD_i}$ . If valid, generate random secret $rs_{GN_j} \in Z_q^*$ , current timestamp $TS_{GN_j}$ . Calculate $RS_{GN_j} = h(TC_{GN_j}    rs_{GN_j}    pr_{GN_j}    RID_{GN_j}    TS_{GN_j}) \cdot G_k$ , $DHK_{GN_j, SD_i} = h(TC_{GN_j}    rs_{GN_j}    pr_{GN_j}    RID_{GN_j}    TS_{GN_j}) \cdot RS_{SD_i}$ , $g_j(RID_{GN_j}, RID_{SD_i}), y_{GN_j} = h(g_j(RID_{GN_j}, RID_{SD_i})    Sig_{SD_i}    TS_{GN_j}) + h(DHK_{GN_j, SD_i}    TS_{SD_i}    TS_{GN_j})$ , $Sig_{GN_j} = h(TC_{GN_j}    rs_{GN_j}    pr_{GN_j}    RID_{GN_j}    TS_{GN_j}) + h(RID_{GN_j}    RID_{SD_i}    Pub_{GN_j}    DHK_{GN_j, SD_i}    y_{GN_j}) * pr_{GN_j} \pmod{q}$ . $Msg_{AC2} = \{RID_{GN_j}, Sig_{GN_j}, RS_{GN_j}, y_{GN_j}, TS_{GN_j}\}$ (via open channel)
Check if $ TS_{GN_j}^* - TS_{GN_j}  < \Delta T$ ? If valid, retrieve $Pub_{GN_j}$ corresponding to $RID_{GN_j}$ . Calculate $DHK_{SD_i, GN_j} = h(TC_{SD_i}    rs_{SD_i}    pr_{SD_i}    RID_{SD_i}    TS_{SD_i}) \cdot RS_{GN_j}$ , $z_{SD_i} = y_{GN_j} \oplus h(DHK_{SD_i, GN_j}    TS_{SD_i}    TS_{GN_j})$ , $Sig_{GN_j} \cdot G_k = RS_{GN_j} + h(RID_{GN_j}    RID_{SD_i}    Pub_{GN_j}    DHK_{SD_i, GN_j}    y_{GN_j}) \cdot Pub_{GN_j}$ ? If valid, generate current timestamp $TS'_{SD_i}$ . Compute $SK_{SD_i, GN_j} = h(DHK_{SD_i, GN_j}    z_{SD_i})$ , $SKV_{SD_i, GN_j} = h(SK_{SD_i, GN_j}    TS'_{SD_i})$ . $Msg_{AC3} = \{SKV_{SD_i, GN_j}, TS'_{SD_i}\}$ (via open channel)	Checks if $ TS_{SD_i}^{**} - TS'_{SD_i}  < \Delta T$ ? If valid, calculate $SK_{GN_j, SD_i} = h(DHK_{GN_j, SD_i}    h(g_j(RID_{GN_j}, RID_{SD_i})    Sig_{SD_i}    TS_{GN_j}))$ , $SKV_{GN_j, SD_i} = h(SK_{GN_j, SD_i}    TS'_{SD_i})$ . Verify if $SKV_{GN_j, SD_i} = SKV_{SD_i, GN_j}$ ? If valid, session key is valid and $SK_{GN_j, SD_i} (= SK_{SD_i, GN_j})$ .
Key Management Phase	
Gateway node ( $GN_j$ )	Edge server ( $ES_l$ )
Create random secret $rs_{GN_{j1}} \in Z_q^*$ , current timestamp $TS_{GN_{j1}}$ . Calculate $RS_{GN_{j1}} = h(RID_{GN_j}    rs_{GN_{j1}}    TC_{GN_j}    TS_{GN_{j1}}    pr_{GN_j}) \cdot G_k$ , $Sig_{GN_{j1}} = h(RID_{GN_j}    rs_{GN_{j1}}    TC_{GN_j}    TS_{GN_{j1}}    pr_{GN_j}) + h(RID_{GN_j}    Pub_{GN_j}    Pub_{ES_l}    TS_{GN_{j1}}) * pr_{GN_j} \pmod{q}$ . $Msg_{KM1} = \{RID_{GN_j}, RS_{GN_{j1}}, Sig_{GN_{j1}}, TS_{GN_{j1}}\}$ (via open channel)	Obtain registration credentials from the blockchain center ( $BC$ ). Check validity of timestamp $TS_{GN_{j1}}$ . If valid, further verify signature $Sig_{GN_{j1}}$ . If valid, create random $rs_{ES_{l2}} \in Z_q^*$ , current timestamp $TS_{ES_{l2}}$ . Calculate $RS_{ES_{l2}} = h(RID_{ES_l}    pr_{ES_l}    rs_{ES_{l2}}    TS_{ES_{l2}}) \cdot G_k$ , $DHK_{ES_l, GN_j} = h(RID_{ES_l}    pr_{ES_l}    rs_{ES_{l2}}    TS_{ES_{l2}}) \cdot RS_{GN_{j1}}$ . Obtain $g_j(RID_{ES_l}, RID_{GN_j})$ , and compute $SK_{ES_l, GN_j} = h(DHK_{ES_l, GN_j}    g_j(RID_{ES_l}, RID_{GN_j}))$ , and signature $Sig_{ES_{l2}} = h(RID_{ES_l}    pr_{ES_l}    rs_{ES_{l2}}    TS_{ES_{l2}}) + h(Pub_{ES_l}    SK_{ES_l, GN_j}    TS_{ES_{l2}}) * pr_{ES_l} \pmod{q}$ . $Msg_{KM2} = \{RID_{ES_l}, RS_{ES_{l2}}, Sig_{ES_{l2}}, TS_{ES_{l2}}\}$ (via open channel)

FIGURE 3. Summary of access control and key management phases

Block Header	
Block Version	$BlkV$
Previous Block Hash	$PreBH$
Merkle Tree Root	$MrkTR$
Industry Type	$IT_m, m = 1, 2, \dots, w$ (Type of an IIoT applications)
Timestamp	$TS_{ES_i}$
Creator of Block	$CreB_{ID}$ (Identity of an $ES_i$ in P2P network)
Public Key of Signer $ES_i$	$Pub_{ES_i}$
Block Payload (Encrypted Transactions)	
List of $t_n$ Encrypted Transactions $\#i(Tx_i)$	$\{E_{Pub_{ES_i}}(Tx_i)   i = 1, 2, \dots, t_n\}$
Current Block Hash	$CurBH$
Signature on $CurBH$	$Sig_{Block_k} = ECDSA.Sig_{pr_{ES_i}}(CurBH)$

FIGURE 4. Architecture of a block  $Block_k$  for various transactions

#### IV. SECURITY ANALYSIS

In this section, we discourse the security analysis to show that the proposed PBACS-PECIIoT is resilient against the following potential attacks.

##### A. FORMAL SECURITY ANALYSIS UNDER ROR MODEL

In this section, we discuss about the “session key security under broadly-recognized Real-Or-Random (ROR) oracle model [61] to show that PBACS-PECIIoT is secure against an adversary  $\mathcal{A}$  for deriving the session-key between a smart device ( $SD_i$ ) and a gateway node ( $GN_j$ ) during the access control phase” described in Section III-B. It is worth noticing that the “ROR-model based security analysis” provides the semi-formal security proof where the advantage of an adversary, say  $\mathcal{A}$ , is computed, and  $\mathcal{A}$  attempts to derive the session key among two communicating entities in the network.

###### 1) Random Oracle Model

We first describe the respective security model that is based on the works by Bellare *et al.* [62] and Wu *et al.* [63], for the proposed scheme, that goes through a sequence of the interactive games between a challenger and an adversary. Here, the main intention is to prove that the proposed scheme provides the session key security against the adversary.

The adversary  $\mathcal{A}$  is permitted to execute the following queries for deriving the session key:

- $Execute(\Theta_{SD_i}^{a_1}, \Theta_{GN_j}^{a_2})$ :  $\mathcal{A}$  carries out this query to eavesdrop the messages exchanged between  $SD_i$  and  $GN_j$ .
- $CorruptSD(\Theta_{SD_i}^{a_1})$ : It allows  $\mathcal{A}$  to extract “the credentials stored in a stolen or lost  $SD_i$ 's memory”.
- $Reveal(\Theta^a)$ : By executing this query, the session key  $SK_{SD_i, GN_j}$  ( $= SK_{GN_j, SD_i}$ ) is exposed to  $\mathcal{A}$  that is shared between  $\Theta^a$  and its respective associate.
- $Test(\Theta^a)$ :  $\mathcal{A}$  is allowed to perform  $\Theta^a$  to verify if the session key  $SK_{SD_i, GN_j}$  ( $= SK_{GN_j, SD_i}$ ) is real or a random key.

Definition 1 of the semantic security is used to show the session key security of PBACS-PECIIoT in Theorem 1. In addition, as discussed in [64], a “collision-resistant one-way cryptographic hash function  $h(\cdot)$  is accessed to all the

**Algorithm 1** Voting-based consensus for verification and addition of a block ( $Block_k$ )

**Input:**  $Block_k = \{\text{Block Header, Block Payload, } CurBH, Sig_{Block_k}\}$ ; private-public key pairs ( $pr_{ES_i}, Pub_{ES_i}$ ), and  $n_{f_{ES_i}}$  represents the number of faulty nodes in P2P network.

**Output:** Commitment & addition of  $Block_k$  in the blockchain after successful validation.

- 1: Assume a leader has been selected by  $ES_i$ , say  $L$  using the leader selection algorithm [41].
- 2:  $L$  generates current timestamp  $TS_L$  and a random number  $r_L$  to perform voting process.  $L$  creates signature  $Sig_L$  using ECDSA signature generation algorithm with its own private key  $pr_L$  on the message  $h(Block_k || TS_L || r_L || VTR_{req})$ , where  $VTR_{req}$  is voting request.
- 3:  $L$  sends the request message  $\langle Block_k, Sig_L, E_{Pub_{ES_i}}[r_L, VTR_{req}], TS_L \rangle$  to each other edge server  $ES_i$  via public channel.
- 4: After receiving request message, each  $ES_i$  checks timestamp  $TS_L$  and if it is valid, it computes  $(r_L, VTR_{req}) = D_{pr_{ES_i}}[E_{Pub_{ES_i}}[r_L, VTR_{req}]]$ , verifies  $Sig_L$  using ECDSA signature verification algorithm [39].
- 5: If the signature is valid,  $ES_i$  further verifies  $MrkTR$ ,  $CurBH$ , and  $Sig_{Block_k}$  on received block  $Block_k$ .
- 6: After all the successful validations,  $ES_i$  sends the response message  $\langle E_{Pub_L}[r_L, TS_{ES_i}, VTR_{res}], TS_{ES_i} \rangle$  to  $L$  via public channel, where  $VTR_{res}$  is the voting response and  $TS_{ES_i}$  is current timestamp.
- 7: Assume that  $VCount$  represents the number of valid votes. Set  $VCount \leftarrow 0$ .
- 8: **for** each received message  $\langle E_{Pub_L}[r_L, TS_{ES_i}, VTR_{res}], TS_{ES_i} \rangle$  from other edge nodes  $ES_i$  **do**
- 9:      $L$  first checks validity of timestamp  $TS_{ES_i}$ , decrypts message using its private key  $pr_L$  and validates  $r_L, TS_{ES_i}$  and  $VTR_{res}$ . If all are valid, set  $VCount = VCount + 1$ .
- 10: **end for**
- 11: **if**  $(VCount > 2n_{f_{ES_i}} + 1)$  **then**
- 12:      $L$  sends commit response for successful verification of  $Block_k$  to its all followers  $ES_i$ .
- 13:     Each  $ES_i$  and  $L$  then add  $Block_k$  to their local ledgers.
- 14: **end if**

FIGURE 5. Voting-based consensus for verification and addition of a block ( $Block_k$ )

involved participants including the adversary  $\mathcal{A}$ . As a result, we also model “ $h(\cdot)$  as a random oracle, say *hash*”. The ROR model is associated with the following components:

**Participants.** As we consider the access control between smart device  $SD_i$  and gateway node  $GN_j$  mentioned in Section III-B, two participants, namely  $SD_i$  and  $GN_j$  are engaged for communication, and apart from these entities the registration authority  $RA_k$  is also involved during offline registration purpose and dynamic node addition phase. The notations  $\Theta_{SD_i}^{a_1}$  and  $\Theta_{GN_j}^{a_2}$  signify the  $a_1^{th}$  and  $a_2^{th}$  instances of  $SD_i$  and  $GN_j$ , respectively. These instances are known as the “random oracles”.

**Accepted state.** An instance  $\Theta^a$  will enter in its “accepted state” once it goes to an accept state when the last valid protocol message is received. If all the communicated messages (sent and received) are put in an ordered sequence, it creates a “session identification  $sid$  of  $\Theta^a$  for the current session”.

**Partnering.** Two instances ( $\Theta^{a_1}$  and  $\Theta^{a_2}$ ) will be the partners to each other if the following are fulfilled: a)  $\Theta^{a_1}$  and  $\Theta^{a_2}$  are in “accepted states”; b)  $\Theta^{a_1}$  and  $\Theta^{a_2}$  share the

same  $sid$  and also “mutually authenticate each other”; and c)  $\Theta^{a_1}$  and  $\Theta^{a_2}$  are “mutual partners of each other”.

**Freshness.** An instance  $\Theta_{SD_i}^{a_1}$  or  $\Theta_{GN_j}^{a_2}$  is *fresh* if the established session key  $SK_{SD_i,GN_j}$  ( $= SK_{GN_j,SD_i}$ ) shared between  $SD_i$  and  $GN_j$  is not revealed to  $\mathcal{A}$  using the  $\text{Reveal}(\Theta^a)$  query described above.

We now define the “semantic security” in Definition 1 prior to prove Theorem 1.

**Definition 1** (Semantic security). *The “advantage of an adversary  $\mathcal{A}$  running in polynomial time  $t$  in breaking the semantic security of the proposed PBACS-PECIIoT for deriving the session key  $SK_{SD_i,GN_j}$  ( $= SK_{GN_j,SD_i}$ ) among a smart device  $SD_i$  and a gateway node  $GN_j$ ” in a particular session during the access control phase (ACP) is  $Adv_{\mathcal{A},ACP}^{PBACS-PECIIoT}(t) = |2Pr[c' = c] - 1|$ , where “ $c$  and  $c'$  are respectively the correct and guessed bits”.*

## 2) Provable Security

In this section, we apply the random oracle model discussed above in order to prove that the proposed scheme provides the session key security that is described in Theorem 1.

**Theorem 1.** *The advantage  $Adv_{\mathcal{A},ACP}^{PBACS-PECIIoT}(t)$  of an adversary  $\mathcal{A}$  running in polynomial time  $t$  in order to derive the session key  $SK_{SD_i,GN_j}$  ( $= SK_{GN_j,SD_i}$ ) established between  $SD_i$  and  $GN_j$  in a particular session during the access control phase (ACP) for the proposed PBACS-PECIIoT is  $Adv_{\mathcal{A},ACP}^{PBACS-PECIIoT}(t) \leq \frac{q_h^2}{|\text{hash}|} + 2Adv_{\mathcal{A}}^{ECDDHP}(t)$ , where  $q_h$ ,  $|\text{hash}|$ , and  $Adv_{\mathcal{A}}^{ECDDHP}(t)$  represent the “number of hash queries”, the “range space of a one-way collision-resistant hash function  $h(\cdot)$ ”, and the “advantage of breaking the Elliptic Curve Decisional Diffie-Hellman Problem (ECDDHP)”, respectively.*

*Proof.* A similar proof is followed here as in [64]. In the proposed PBACS-PECIIoT, we consider three games, namely  $Game_i^A$  for the adversary  $\mathcal{A}$ ,  $i = 0, 1, 2$ . We define  $Succ_{Game_i}^A$  as an event wherein  $\mathcal{A}$  can guess the random bit  $c$  correctly in the game  $Game_i^A$ . Therefore,  $\mathcal{A}$ 's advantage to win the  $Game_i^A$  in the proposed PBACS-PECIIoT becomes  $Adv_{\mathcal{A},Game_i}^{PBACS-PECIIoT} = Pr[Succ_{Game_i}^A]$ . The games are now defined as follows.

**Game<sub>0</sub><sup>A</sup>:** Under this game, the adversary  $\mathcal{A}$  plays a real attack under the ROR model for the initial game  $Game_0^A$ . Prior to beginning of the game  $Game_0^A$ ,  $\mathcal{A}$  needs to pick a random bit  $c$ . Therefore, the advantage of  $Game_0^A$  is then

$$Adv_{\mathcal{A},ACP}^{PBACS-PECIIoT}(t) = |2Adv_{\mathcal{A},Game_0}^{PBACS-PECIIoT} - 1|. \quad (1)$$

**Game<sub>1</sub><sup>A</sup>:** In this game,  $\mathcal{A}$  applies the eavesdropping attack to derive the session key for a particular session.  $\mathcal{A}$  performs the *Execute* query to intercept all the communicated messages  $Msg_{AC1} = \{RID_{SD_i}, Sig_{SD_i}, RS_{SD_i}, TS_{SD_i}\}$ ,  $Msg_{AC2} = \{RID_{GN_j}, Sig_{GN_j}, RS_{GN_j}, y_{GN_j}, TS_{GN_j}\}$  and  $Msg_{AC3} = \{SKV_{SD_i,GN_j}, TS'_{SD_i}\}$  during the access control phase (ACP) between  $SD_i$  and  $GN_j$  mentioned in

Section III-B. After that,  $\mathcal{A}$  may try to generate the session key  $SK_{SD_i,GN_j} = h(DHK_{SD_i,GN_j} || z_{SD_i})$ , where  $DHK_{SD_i,GN_j} = h(TC_{SD_i} || rs_{SD_i} || pr_{SD_i} || RID_{SD_i} || TS_{SD_i}) \cdot RS_{GN_j}$  and  $z_{SD_i} = y_{GN_j} \oplus h(DHK_{SD_i,GN_j} || TS_{SD_i} || TS_{GN_j})$ . Without knowledge of the long term secrets  $\{TC_{SD_i}, pr_{SD_i}\}$  and  $\{TC_{GN_j}, pr_{GN_j}\}$ ,  $\mathcal{A}$  cannot succeed to derive the session key  $SK_{SD_i,GN_j}$  ( $= SK_{GN_j,SD_i}$ ). As the credentials are protected by the “cryptographic hash function  $h(\cdot)$ ”,  $\mathcal{A}$  will be unable to derive the session key even by executing the *Reveal* and *Test* queries. Therefore, the games  $Game_1^A$  and  $Game_0^A$  are both indistinguishable under such an eavesdropping attack. The following outcome is then produced:

$$Adv_{\mathcal{A},Game_1}^{PBACS-PECIIoT} = Adv_{\mathcal{A},Game_0}^{PBACS-PECIIoT}. \quad (2)$$

**Game<sub>2</sub><sup>A</sup>:** In this game, the adversary  $\mathcal{A}$  plays an active attack.  $\mathcal{A}$  simulates the *hash* and *CorruptSD* queries and tries to solve computational ECDDHP problem.  $\mathcal{A}$  needs to obtain  $DHK_{SD_i,GN_j} = h(TC_{SD_i} || rs_{SD_i} || pr_{SD_i} || RID_{SD_i} || TS_{SD_i}) \cdot RS_{GN_j}$  ( $= DHK_{GN_j,SD_i}$ ) to derive the session key  $SK_{SD_i,GN_j}$ . Assume that  $\mathcal{A}$  hijacks all the transmitted message  $\{Msg_{AC1}, Msg_{AC2}, Msg_{AC3}\}$ . Thus,  $\mathcal{A}$  knows the values  $RS_{SD_i}$  and  $RS_{GN_j}$ . From  $RS_{SD_i}$  and  $RS_{GN_j}$ ,  $\mathcal{A}$  may try to compute the secret values  $h(TC_{SD_i} || rs_{SD_i} || pr_{SD_i} || RID_{SD_i} || TS_{SD_i})$  and  $h(TC_{GN_j} || rs_{GN_j} || pr_{GN_j} || RID_{GN_j} || TS_{GN_j})$ , respectively. However, to derive these secrets credentials,  $\mathcal{A}$  needs to know the long term secrets  $\{TC_{SD_i}, pr_{SD_i}, TC_{GN_j}, pr_{GN_j}\}$ , which becomes difficult problem due to solving ECDDHP. Moreover, the secrets are enclosed by a “one-way collision-resistant hash function  $(h(\cdot))$ ”. In addition,  $\mathcal{A}$  will execute *CorruptSD* to extract all the secret credentials  $\{RID_{SD_i}, TC_{SD_i}, (pr_{SD_i}, Pub_{SD_i}), h(\cdot), E_q(\alpha, \beta), G_k\}$ , but he/she has no knowledge about the random secrets (short term secrets) $\{rs_{SD_i}, rs_{GN_j}\}$ . If  $\mathcal{A}$  is aware of the long term secrets as well as short term, then only he/she gets the session key  $SK_{SD_i,GN_j}$  ( $= SK_{GN_j,SD_i}$ ). Therefore, the games  $Game_2^A$  and  $Game_1^A$  are indistinguishable if we exclude the *hash* and *CorruptSD* queries in  $Game_2^A$ . The birthday paradox result on “one-way collision-resistant hash function  $(h(\cdot))$ ” and ECDDHP will result in the following relation:

$$\begin{aligned} |Adv_{\mathcal{A},Game_1}^{PBACS-PECIIoT} - Adv_{\mathcal{A},Game_2}^{PBACS-PECIIoT}| \\ \leq \frac{q_h^2}{2|\text{hash}|} + Adv_{\mathcal{A}}^{ECDDHP}(t). \end{aligned} \quad (3)$$

Since all the games have been executed by  $\mathcal{A}$ , and it is “only remaining for  $\mathcal{A}$  to correctly guess a bit to win the game  $Game_2^A$ ”, we have,

$$Adv_{\mathcal{A},Game_2}^{PBACS-PECIIoT} = \frac{1}{2}. \quad (4)$$

Eq. (1) gives

$$\frac{1}{2} \cdot Adv_{\mathcal{A},ACP}^{PBACS-PECIIoT}(t) = |Adv_{\mathcal{A},Game_0}^{PBACS-PECIIoT} - \frac{1}{2}|. \quad (5)$$

Eq. (3) leads to the following inequality using Eq. (5):

$$\begin{aligned} & \frac{1}{2} \cdot \text{Adv}_{\mathcal{A}, \text{ACP}}^{\text{PBACS-PECIIoT}}(t) \\ = & |\text{Adv}_{\mathcal{A}, \text{Game}_0}^{\text{PBACS-PECIIoT}} - \text{Adv}_{\mathcal{A}, \text{Game}_2}^{\text{PBACS-PECIIoT}}| \\ = & |\text{Adv}_{\mathcal{A}, \text{Game}_1}^{\text{PBACS-PECIIoT}} - \text{Adv}_{\mathcal{A}, \text{Game}_2}^{\text{PBACS-PECIIoT}}| \\ \leq & \frac{q_h^2}{2|\text{hash}|} + \text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t). \end{aligned} \quad (6)$$

Hence, we have the final result:  $\text{Adv}_{\mathcal{A}, \text{ACP}}^{\text{PBACS-PECIIoT}}(t) \leq \frac{q_h^2}{|\text{hash}|} + 2\text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t)$ .  $\square$

**Remark 4.** If  $\text{Adv}_{\mathcal{A}, \text{KMP}}^{\text{PBACS-PECIIoT}}(t)$  be the advantage of an adversary  $\mathcal{A}$  running in polynomial time  $t$  in order to derive the pairwise secret key  $\text{SK}_{GN_j, ES_l}$  ( $= \text{SK}_{ES_l, GN_j}$ ) established between  $GN_j$  and  $ES_l$  in a particular session during the key management phase (KMP) for the proposed PBACS-PECIIoT, similar to Theorem 1, we also have:  $\text{Adv}_{\mathcal{A}, \text{KMP}}^{\text{PBACS-PECIIoT}}(t) \leq \frac{q_h^2}{|\text{hash}|} + 2\text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t)$ .

## B. INFORMAL SECURITY ANALYSIS

### 1) Replay Attack

In PBACS-PECIIoT, during the access control phase described in Section III-B between a smart device  $SD_i$  and its gateway node  $GN_j$ , the communicated messages  $Msg_{AC1}$ ,  $Msg_{AC2}$ , and  $Msg_{AC3}$  have both random nonces and current timestamps. The freshness of the messages is provided by checking the timestamps. Similarly, for the key management among  $GN_j$  and its associated edge server  $ES_l$  described in Section III-C the communicated messages  $Msg_{KM1}$  and  $Msg_{KM2}$  are also having random numbers and current timestamps. Thus, the receivers can easily detect the old replayed messages that are re-transmitted by an adversary by validating the attached timestamps of the messages. Therefore, PBACS-PECIIoT is resilient against “replay attack”.

### 2) Man-in-the-Middle(MiTM) Attack

Suppose an adversary  $\mathcal{A}$  eavesdrops the access control request message  $Msg_{AC1} = \{RID_{SD_i}, Sig_{SD_i}, RS_{SD_i}, TS_{SD_i}\}$  and tries to send another valid message, say  $Msg_{AC1}^*$  to the receiver  $GN_j$ . To achieve this goal,  $\mathcal{A}$  can select a random number  $rs_{SD_i}^* \in Z_q^*$  and timestamp  $TS_{SD_i}^*$  on the fly, and then calculate  $RS_{SD_i}^* = h(TC_{SD_i} || rs_{SD_i}^* || pr_{SD_i} || RID_{SD_i} || TS_{SD_i}^*) \cdot G_k$ . Without knowledge of the temporal credential  $TC_{SD_i}$  and permanent secret  $pr_{SD_i}$ ,  $\mathcal{A}$  can not compute valid  $RS_{SD_i}^*$  and other valid signature  $Sig_{SD_i}^*$  for  $Msg_{AC1}^*$ . Similarly, by intercepting the messages  $Msg_{AC2}$ ,  $Msg_{AC3}$ ,  $Msg_{KM1}$  and  $Msg_{KM2}$ , without temporal credentials and permanent secret,  $\mathcal{A}$  can modify them on the fly. PBACS-PECIIoT is then resilient against “MiTM attack”.

### 3) Impersonation Attacks

Assume an adversary  $\mathcal{A}$  plays as a legitimate smart device and tries to communicate with the gateway node by creating

a valid message  $Msg_{AC1} = \{RID_{SD_i}, Sig_{SD_i}, RS_{SD_i}, TS_{SD_i}^*\}$ . For successful attack,  $\mathcal{A}$  can pick a random secret  $rs_{SD_i} \in Z_q^*$  and timestamp  $TS_{SD_i}^*$  to calculate  $RS_{SD_i} = h(TC_{SD_i} || rs_{SD_i} || pr_{SD_i} || RID_{SD_i} || TS_{SD_i}^*) \cdot G_k$ . Since  $\mathcal{A}$  has no idea about secrets  $TC_{SD_i}$  and  $pr_{SD_i}$ ,  $\mathcal{A}$  can not compute valid  $Msg_{AC1}$ . Similarly, it is also a “computationally impossible task” for  $\mathcal{A}$  to construct other valid messages  $Msg_{AC2}$ ,  $Msg_{AC3}$ ,  $Msg_{KM1}$  and  $Msg_{KM2}$ . This means that PBACS-PECIIoT is secure against “smart device, gateway node and edge server impersonation attacks”.

### 4) Privileged-Insider Attack

During the registration phase,  $RA_k$  does registration of all the entities ( $SD_i$ ,  $GN_j$ , and  $ES_l$ ) without providing any registration information from the entities. Instead,  $RA_k$  deletes all the secrets information (for example, temporal credentials and private keys) after the credentials are stored in the memory of the registering parties after successful registration prior to their deployment in a particular IIoT application. An adversary, being a privileged-insider user of any  $RA_K$ , can not then obtain any pre-loaded secret credentials of the deployed entities. Hence, PBACS-PECIIoT is resilient against “privileged-insider attack”.

### 5) Physical IoT Smart Device Capture Attack

Due to existence of an unethical territory, there is a high chance that an adversary  $\mathcal{A}$  can physically capture few IoT smart devices  $SD_i$ , and extract their stored credentials  $\{RID_{SD_i}, TC_{SD_i}, (pr_{SD_i}, Pub_{SD_i}), h(\cdot), E_q(\alpha, \beta), G_k\}$  by applying the “power analysis attacks” [50]. However, the stored credentials are unique and different for all smart devices  $SD_i$ . Therefore, it is not possible for  $\mathcal{A}$  to establish the session keys between a non-compromised  $SD_i$  and its respective  $GN_j$ . This circumstance is known as “unconditionally secure against smart device capture attack”. As a result, PBACS-PECIIoT is secure against “physical vehicle capture attack”.

### 6) Ephemeral Secret Leakage (ESL) Attack

During the access control process between  $SD_i$  and  $GN_j$ , they establish a common session key  $\text{SK}_{SD_i, GN_j} = h(DHK_{SD_i, GN_j} || z_{SD_i})$  ( $= \text{SK}_{GN_j, SD_i}$ ) where  $DHK_{SD_i, GN_j} = h(TC_{SD_i} || rs_{SD_i} || pr_{SD_i} || RID_{SD_i} || TS_{SD_i}) \cdot RS_{GN_j}$ . Similarly, during key management phase between  $GN_j$  and  $ES_l$ , a common session key is established as  $\text{SK}_{GN_j, ES_l} = h(DHK_{GN_j, ES_l} || g_j(RID_{GN_j}, RID_{ES_l}))$  ( $= \text{SK}_{ES_l, GN_j}$ ), where  $DHK_{GN_j, ES_l} = h(RID_{GN_j} || rs_{GN_j} || TC_{GN_j} || TS_{GN_j} || pr_{GN_j}) \cdot RS_{ES_l}$ . In both the scenarios, in order to calculate  $DHK_{SD_i, GN_j}$  and  $DHK_{GN_j, ES_l}$  the short term (random nonces) and long term secrets (temporal credentials and private keys) are necessary. Since in every session the session keys are unique and distinct, even through a session key is compromised in a particular session it does not affect on the session (secret) keys established in other sessions. PBACS-PECIIoT is then secure against “session-temporary information attack” and

it also provides the “perfect forward and backward secrecy” goals at the same time.

#### 7) Block Verification in Blockchain

In PBACS-PECIIoT, suppose a verifier  $\mathcal{V}$  wants to verify a given block, say  $Block_k$  in the blockchain. To successfully verify  $Block_k$ ,  $\mathcal{V}$  requires computation of “Merkle tree root ( $MrkTR$ )” on encrypted transactions and “current block hash ( $CurBH$ )” on all the entities in  $Block_k$ . If  $MrkTR^* = MrkTR$  and  $CurBH^* = CurBH$ ,  $\mathcal{V}$  further validates  $Sig_{Block_k}$  using “ECDSA signature verification algorithm” with the public key  $Pub_{ES_l}$  of  $ES_l$ . Since  $\mathcal{V}$  verifies all the  $MrkTR$ ,  $CurBH$  and  $Sig_{Block_k}$ , it is quite hard for an adversary to tamper the block  $Block_k$  in the blockchain. If all the validations are successful,  $\mathcal{V}$  accepts  $Block_k$  as a valid block in the blockchain.

#### 8) Transaction Privacy Leakage

In blockchain, the user behavior can be traceable and it is important to preserve the transaction privacy of the users. A transaction in public blockchain may contain sensitive information and leakage of such critical data is a serious concern. Also, it is important to note that the input transaction should not be linked to its corresponding outputs. The “Bitcoin” and “Zcash” use one-time account to receive cryptograms/puzzles. A secret key of user can be used within it so that an attacker cannot derive whether the same transaction contains a user’s credential. Moreover, a common wallet may also leakage some vital information of the user. In the proposed PBACS-PECIIoT, due to the private blockchain criteria, the transactions in a block are encrypted with the help of public key of the corresponding edge server  $Pub_{ES_l}$ . Therefore, the privacy of the transactions are fulfilled in PBACS-PECIIoT.

#### 9) Selfish Mining Attack

Selfish mining attack is introduced by Eyal *et al.* in 2014 [65]. In a selfish mining attack [65], an attacker may misuse the computation power and steal the inappropriate rewards from the legitimate miners [66]. The attackers in the selfish mining may aim to retain the large private chain as compared to the public branch so that they can individually hold and dominate to add the additional new blocks. Thus, the selfish miners can obtain more blocks and have a competitive advantage over legitimate miners. This strategy has been extensively mentioned in Bitcoin, but very few attentions have been given to address it. Davidson *et al.* [67] mentioned how the selfish mining can increment the earning of the miners for a larger collection of cryptocurrencies. In PBACS-PECIIoT, we have considered private blockchain and the mining is done by the P2P edge nodes which are treated as semi-trusted. Therefore, selfish mining attack would be hard to perform in the proposed system.

#### 10) Balance Attack

In this attack, an attacker tries to introduce a delay network communication between a valid range of subgroups consisting of similar mining power capabilities to execute the transactions. However, the miner needs to mine sufficient blocks to assure the subtree of another subgroup is equally essential as compared to the transaction subgroups. Moreover, an attacker can collect the transactions, which are not committed, in order to form a block and it has immense possibility of exceeding the subtree which consists of the transactions. In PBACS-PECIIoT, the individual edge server is connected with each application and it is semi-trusted in the private blockchain. As a result, it is difficult to create a separate chain and mine sufficient blocks into the blockchain. Hence, the balance attack is eliminated in the proposed PBACS-PECIIoT.

#### 11) Sybil Attack

In this attack, an attacker can damage the reputation system by forging the identities (i.e. fake users’ accounts) in the P2P network and use them to achieve the extremely huge domination in the network for making the legitimate entities in minority. Such virtual nodes or illegitimate nodes can then perform like genuine nodes to establish disproportionately huge influence on the P2P network. These may lead to various other attacks, such as “Denial-of-Service (DoS)” and “Distributed Denial-of-Service (DDoS)” attacks. However, it is required to verify or authenticate such nodes and the identities prior to joining the network. In PBACS-PECIIoT, if an edge server behaves like an attacker and tries to perform Sybil attack, it can not dominate the entire network and make the legitimate entities in minority. Therefore, the Sybil attack is resisted in the proposed PBACS-PECIIoT.

### V. FORMAL SECURITY VERIFICATION USING AVISPA: SIMULATION STUDY

The “AVISPA tool (Automated Validation of Internet Security Protocols and Applications)” is a push-button software validation tool that provides a “modular and expressive formal language for specifying security protocols and properties, known as the High-Level Protocol Specification Language (HLPSL)” and integrates various back-ends which help in implementing a “variety of automatic analysis techniques ranging from protocol falsification (by finding an attack on the input protocol) to abstraction-based verification methods for infinite numbers of sessions” [68]. AVISPA contains four backends, namely a) “On-the-Fly Model-Checker (OFMC)”, b) “Constraint-Logic-based Attack Searcher (CL-AtSe)”, c) “SAT-based Model Checker (SATMC)” and d) “Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP)”. Details on AVISPA tool and its associated HLPSL specifications can be referred to the readers in [69].

The proposed scheme (PBACS-PECIIoT) has been implemented under the HLPSL for two scenarios:

SUMMARY	
<b>SAFE</b>	
DETAILS	
BOUNDED_NUMBER_OF_SESSIONS	
PROTOCOL	
/home/sourav/Desktop/span	
/testsuite/results/case1.if	
GOAL as specified	
BACKEND OFMC	
STATISTICS	
TIME 1659 ms	
parseTime 0 ms	
visitedNodes: 1550 nodes	
depth: 9 plies	
a) Case 1 simulation results	
SUMMARY	
<b>SAFE</b>	
DETAILS	
BOUNDED_NUMBER_OF_SESSIONS	
PROTOCOL	
/home/sourav/Desktop/span	
/testsuite/results/case2.if	
GOAL as specified	
BACKEND OFMC	
STATISTICS	
TIME 442 ms	
parseTime 0 ms	
visitedNodes: 272 nodes	
depth: 8 plies	
b) Case 2 simulation results	

FIGURE 6. Simulation results of PBACS-PECIIoT (cases 1 and 2)

- **Case 1:** It implements the registration and access control phases
- **Case 2:** It implements the registration and key management phases

In both the cases, we have basic roles and the mandatory roles for the session and also for the goal and environment. Since AVISPA implements the DY threat model [52] (as discussed in our threat model in Section 2.2), an intruder (*i*) always takes part of an active participating entity during the communication. Due to this, AVISPA has the ability to check whether a tested security protocol is resilient against “replay attack” and “man-in-the-middle attack”. We have simulated both the cases of PBACS-PECIIoT using the “SPAN, the Security Protocol ANimator for AVISPA” [70] under the widely-used OFMC backend. The simulation results demonstrated in Figure 6 clearly show that PBACS-PECIIoT is robust against both replay and man-in-the-middle attacks.

## VI. EXPERIMENTS USING MIRACL

We have done the testbed experiments for various cryptographic primitives with the help of widely-accepted “Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL)” [49]. MIRACL is a C/C++ based Crypto SDK which is regarded by the software developers and cryptographers as the “gold standard open source SDK for elliptic curve cryptography (ECC)”.

We use the following cryptographic primitives for the testbed experiments. The notations  $T_h$ ,  $T_{ecm}$ ,  $T_{eca}$ ,  $T_{me}$ ,  $T_{bp}$ ,  $T_m$ ,  $T_a$ , and  $T_{senc}/T_{sdec}$  denote the time required to execute a “one-way cryptographic hash function”, an “elliptic curve point (scalar) multiplication”, an “elliptic curve point addition”, a “modular exponentiation operation”, a “bilinear pairing operation”, a “modular multiplication over  $GF(q)$ ”, a “modular addition over  $GF(q)$ ”, and a “symmetric encryption/decryption”, respectively. We also considered a non-singular elliptic curve of the type:  $y^2 = x^3 + \alpha x + \beta \pmod{q}$  for “elliptic curve point addition and multiplication”.

In the following, we consider the following two scenarios:

- The first platform that we have considered is for a server and the environment setting as “Ubuntu 18.04.4 LTS, with memory: 7.7 GiB, processor: Intel Core i7-8565U CPU @ 1.80GHz × 8, OS type: 64-bit and disk: 966.1 GB”.
- In the second platform, we have considered a smart device under the Raspberry PI 3 implementation where the environment setting is “Raspberry PI 3 B+ Rev 1.3, with CPU: 64-bit, Processor: 1.4 GHz Quad-core, 4 cores, Memory (RAM): 1GB, and OS: Ubuntu 20.04 LTS, 64-bit” [71].

All the experiments are run for 100 times for each cryptographic primitive under both the platforms, and we have then considered the “maximum, minimum and average runtime (in milliseconds) for each cryptographic primitive”. The experimental results for various cryptographic primitives under a server platform and under the Raspberry PI 3 setting are provided in Table 3.

TABLE 3. Experimental results of cryptographic primitives on a server and a Raspberry PI 3 using MIRACL

Primitive	Average time (ms) under sever	Average time (ms) under Raspberry PI 3
$T_{ecm}$	0.674	2.288
$T_{eca}$	0.002	0.016
$T_h$	0.055	0.309
$T_{me}$	0.072	0.228
$T_m$	0.002	0.011
$T_a$	0.001	0.010
$T_{bp}$	4.716	32.084
$T_{senc}$	0.001	0.018
$T_{sdec}$	0.001	0.014

It is worth noticing that a Raspberry PI uses a “micro SD card” that has the capability to store both the system and data. If we compare a “micro SD card” to the “modern hard drives or solid-state drive (SSD) that are commonly found in computers (Desktops or Laptops)”, the operations like reading and writing on the card are then quite slow in case of Raspberry PI [72]. This is why the results reported

in Table 3 show the average time difference between the server and Raspberry PI. It is worth noticing that these experimental results are used for our comparative study with respect to computational costs for various schemes including the proposed scheme (see Section VII-A).

## VII. COMPARATIVE STUDY

In this section, we provide a detailed comparative study on “security and functionality features”, “communication costs” and “computation costs” among the proposed PBACS-PECIIoT and other state-of-art schemes of Li *et al.* [21], Luo *et al.* [25], Xue *et al.* [23], Garg *et al.* [36].

### A. COMMUNICATION COSTS COMPARISON

In PBACS-PECIIoT, to evaluate the communication costs for the access control phase (Case 1) between  $SD_i$  and  $GN_j$  and for the key management phase (Case 2) among  $GN_j$  and  $ES_l$ , we consider only communication messages among them. It is assumed that a “random number”, an “identity”, a “one-way hash function (using SHA-256 hashing algorithm)”, an “elliptic curve point  $P \in E_q(\alpha, \beta)$ ” and a “timestamp” are 160, 160, 256, 320, and 32 bits, respectively.

In Case 1 of PBACS-PECIIoT, the communication costs for the messages  $Msg_{AC1} = \{RID_{SD_i}, Sig_{SD_i}, RS_{SD_i}, TS_{SD_i}\}$ ,  $Msg_{AC2} = \{RID_{GN_j}, Sig_{GN_j}, RS_{GN_j}, y_{GN_j}, TS_{GN_j}\}$  and  $Msg_{AC3} = \{SKV_{SD_i, GN_j}, TS'_{SD_i}\}$  require  $(256 + 160 + 320 + 32) = 768$  bits,  $(256 + 160 + 320 + 256 + 32) = 1024$  bits, and  $(256 + 32) = 288$  bits, which altogether demand 2080 bits. In Case 2 of PBACS-PECIIoT, the messages  $Msg_{KM1} = \{RID_{GN_j}, RS_{GN_{j1}}, Sig_{GN_{j1}}, TS_{GN_{j1}}\}$  and  $Msg_{KM2} = \{RID_{ES_l}, RS_{ES_{l2}}, Sig_{ES_{l2}}, TS_{ES_{l2}}\}$  needs equally  $(256 + 320 + 160 + 32) = 768$  bits, which altogether require 1536 bits. The comparative study shown in Table 4 demonstrates that the communication costs for both Case 1 and Case 2 require less costs as compared to other schemes.

**TABLE 4.** Comparison of communication costs

Protocol	No. of messages	Total cost (in bits)
PBACS-PECIIoT (Case 1)	3	2080
PBACS-PECIIoT (Case 2)	2	1536
Li <i>et al.</i> [21]	4	5408
Luo <i>et al.</i> [25]	2	3040
Xue <i>et al.</i> [23]	5	9344
Garg <i>et al.</i> [36]	6	4352

### B. COMPUTATION COSTS COMPARISON

Assume  $T_{poly}$  denotes the time required for “evaluation of an  $t$ -degree uni-variate polynomial”. Based on the Horner’s rule [73], evaluating an “ $t$ -degree uni-variate polynomial” requires “ $t$  modular multiplications” and “ $t$  modular additions”, that is,  $T_{poly} = tT_m + tT_a$ . We have used the average time listed in Table 3 needed for various cryptographic primitives for a server. On the other side, we have used the average time listed in Table 3 needed for various

cryptographic primitives for a smart device or user’s mobile device under Raspberry PI 3.

In Case 1 of PBACS-PECIIoT, an IoT smart device  $SD_i$  requires the computation cost of  $6T_h + 4T_{ecm} + T_{eca} \approx 11.022$  ms and a gateway note  $GN_j$  needs the computation cost of  $7T_h + 4T_{ecm} + T_{eca} + T_{poly} \approx 6.083$  ms. In Case 2 of PBACS-PECIIoT, both  $GN_j$  and  $ES_l$  equally need the computation cost of  $4T_h + 4T_{ecm} + T_{eca} + T_{poly} \approx 5.918$  ms. Here, we have considered  $t = 1000$  to support “unconditional security” as suggested by Blundo *et al.* [59]. The comparative analysis on computation costs in PBACS-PECIIoT for both Case 1 and Case 2 shows that PBACS-PECIIoT needs comparable costs with other existing schemes that are tabulated in Table 5.

**TABLE 5.** Comparative computational costs analysis

Scheme	Smart device/User cost	Server cost (GN/ES)
PBACS-PECIIoT (Case 1)	$6T_h + 4T_{ecm} + T_{eca}$ $\approx 11.022$ ms	$7T_h + 4T_{ecm} + T_{eca}$ $+tT_m \approx 6.083$ ms
PBACS-PECIIoT (Case 2)	—	$8T_h + 8T_{ecm} + 2T_{eca}$ $+2tT_m \approx 11.836$ ms
Li <i>et al.</i> [21]	$5T_{me} + 3T_h$ $\approx 2.067$ ms	$4T_{me} + 3T_h + T_{bp} +$ $T_{ecm} + T_{eca} \approx 5.845$ ms
Luo <i>et al.</i> [25]	$T_{bp} + T_h$ $\approx 32.393$ ms	$3T_{ecm} + 3T_{bp} + 3T_h +$ $T_{eca} + T_{me} \approx 16.409$ ms
Xue <i>et al.</i> [23]	$4T_{ecm} + T_{eca} + 5T_{bp}$ $+4T_h + 6T_{senc}/T_{sdec}$ $\approx 170.92$ ms	$3T_h + 3T_{me} + 2T_{ecm}$ $+T_{eca} + T_{bp}$ $\approx 6.447$ ms
Garg <i>et al.</i> [36]	$6T_{ecm} + 8T_h$ $\approx 16.2$ ms	$6T_{ecm} + 8T_h$ $4.484$ ms

### C. SECURITY AND FUNCTIONALITY FEATURES COMPARISON

Various “security and functionality features” ( $FSF_1 - FSF_{16}$ ) are considered in comparative study among PBACS-PECIIoT and other schemes (see Table 6). It is evident that PBACS-PECIIoT provides better security features and more functionality attributes as compared to those for other schemes of Li *et al.* [21], Luo *et al.* [25] and Xue *et al.* [23]. Considering the comparative analysis on “communication and computation costs” and “security and functionality features” ( $FSF_1 - FSF_{16}$ ), we can say that PBACS-PECIIoT is much practical to be deployed for PEC in IIoT environment.

Since the fog servers are semi-trusted, the distributed databases with only timestamps can not help to fulfill all the security requirements such as insider attack, device physical capture attack, and most importantly session key security (ESL attack) under the CK-adversary model. However, the proposed PBACS-PECIIoT provides the security features as compared to the existing schemes.

## VIII. BLOCKCHAIN IMPLEMENTATION

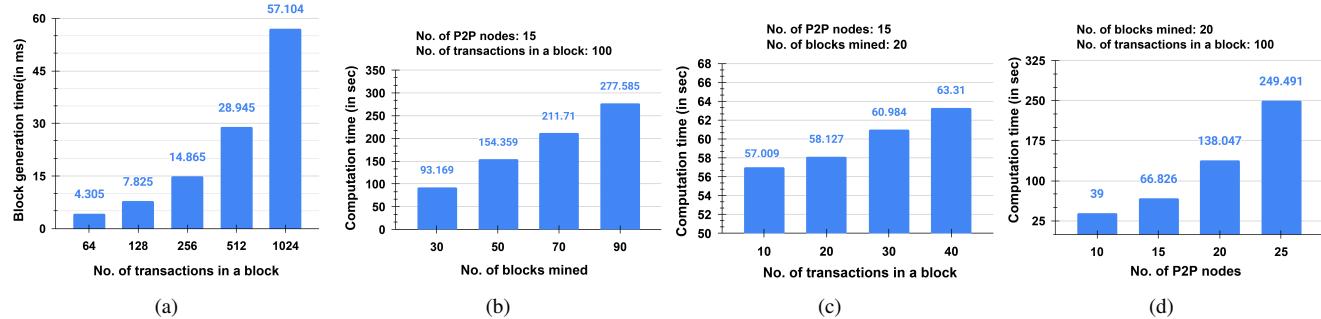
In this section, we present the practical implementation of our proposed PBACS-PECIIoT, and measure its performance in terms of computational time. The computational time is considered to measure the costs for a block addition and mined in the P2P network. The performance evaluation is

**TABLE 6.** Comparison of functionality & security features

Feature	Li <i>et al.</i> [21]	Luo <i>et al.</i> [25]	Xue <i>et al.</i> [23]	Garg <i>et al.</i> [36]	PBACS-PECIIoT
$FSF_1$	✓	✓	✓	✓	✓
$FSF_2$	✓	✓	✓	✓	✓
$FSF_3$	✓	✓	✓	✓	✓
$FSF_4$	✗	✗	✓	✓	✓
$FSF_5$	✓	✓	✗	✓	✓
$FSF_6$	✓	✓	✓	✓	✓
$FSF_7$	✓	✗	✓	✓	✓
$FSF_8$	N/A	✓	N/A	✓	✓
$FSF_9$	✗	✗	✗	✗	✓
$FSF_{10}$	✗	✓	✗	✓	✓
$FSF_{11}$	✗	✗	✗	✗	✓
$FSF_{12}$	✗	✗	✗	✗	✓
$FSF_{13}$	✗	✗	✗	N/A	✓
$FSF_{14}$	✗	✗	✗	N/A	✓
$FSF_{15}$	✗	✗	✗	N/A	✓
$FSF_{16}$	✗	✗	✗	N/A	✓

Note:  $FSF_1$ : “resistant to privileged insider attack”;  $FSF_2$ : “replay attack”;  $FSF_3$ : “man-in-the-middle attack”;  $FSF_4$ : “mutual authentication”;  $FSF_5$ : “key agreement”;  $FSF_6$ : “device/gateway node impersonation attack”;  $FSF_7$ : “resilience against device physical capture attack”;  $FSF_8$ : “edge server impersonation attack”;  $FSF_9$ : “session key security under the CK-adversary model”;  $FSF_{10}$ : “formal security verification using AVISPA tool”;  $FSF_{11}$ : “dynamic node addition phase”;  $FSF_{12}$ : “support to blockchain-based solution”;  $FSF_{13}$ : “transaction privacy leakage”;  $FSF_{14}$ : “selfish mining attack”;  $FSF_{15}$ : “balance attack”;  $FSF_{16}$ : “Sybil attack”.

✓: “a scheme is secure or it assists a feature”; ✗: “a scheme is insecure or it does not assist a feature”; N/A: “not applicable in a scheme”.



**FIGURE 7.** (a) Block generation time (in milliseconds) by an edge server,  $ES_l$  (b) Blockchain simulation results in Case-I (c) Blockchain simulation results in Case-II (d) Blockchain simulation results in Case-III

considered using a reasonable amount of data for simulation. However, the proposed model can also handle a huge volume of data for an IIoT environment. Many discussions are for scalability issues in blockchain, but in real world scenario the lightning network [74] can be used to handle the high transactions volume. The lightning network is a layer 2 protocol which is specifically used to improve the scalability in blockchain network. The environment was considered for simulation with the following setting: “CPU Architecture: 64-bit, Processor: 2.60 GHz Intel Core i5-3230M, Memory: 8 GB, OS: Ubuntu 18.04.4 LTS”.

In each block in the blockchain, we have the block version ( $BlkV$ ), previous block hash ( $PreBH$ ), Merkle tree root ( $MrkTR$ ), industry type ( $IT_m$ ), timestamp ( $TS_{ES_l}$ ), creator of block ( $CreBID$ ), public Key of signer ( $Pub_{ES_l}$ ),

current block hash ( $CurBH$ ), signature ( $Sig_{Block_k}$ ), whose sizes are taken as 32, 256, 256, 32, 32, 160, 320, 256 and 320 bits, respectively. In addition, each encrypted transaction  $E_{Pub_{ES_l}}(Tx_{t_i})$ , ( $i = 1, 2, \dots, t_n$ ), consists of two elliptic curve points and hence, it needs  $(320+320) = 640$  bits. The total block size then turns out to be  $1664 + 640t_n$  bits.

In order to measure the block generation time in the proposed PBACS-PECIIoT with respect to the block structure mentioned in Figure 4, we have considered the average computational time (in milliseconds) for hash function and ECDSA signature generation under the MIRACL library for a server setting platform (see Table 3). This is because each edge server is resource rich node in the network. Note that the time needed for an ECDSA signature generation is approximately  $T_{ecm} + T_h$ . In addition, we have also imple-

mented the Merkle tree using SHA-256 hashing algorithm. Based on these results, an edge server can compute the block generation time. In Figure 7(a), a block generation time (in milliseconds) by an edge server is shown for various number of encrypted transactions containing in the block. The results show that the computational time increases when the number of encrypted transactions in a block also increases.

Now, the blockchain implementation has been performed using the node.js language with VS CODE 2019 with the voting-based consensus algorithm explained in Figure 5. The following three cases are taken:

- **Case-I:** We have considered the number of blocks mined versus the total computational time (in milliseconds) with the number of P2P nodes as 15 and the number of transactions per each block as 100. The blockchain simulation outcomes under this scenario are presented in Figure 7(b). It is observed that “as the number of blocks mined increases, the total computational time increases”.
- **Case-II:** In this case, we have considered the number of transactions in per block versus the total computational time (in milliseconds). The number of blocks mined is fixed at 20, whereas the number of P2P nodes remains as in Case 1 as 15. Figure 7(c) presents the simulation results. It is worth noticing that the “total computational time increases as the number of transactions per block also increases”.
- **Case-III:** In this case, we have considered the number of P2P nodes versus the total computational time (in milliseconds). Moreover, the number of blocks mined is fixed at 20 and the number of transactions in per block is also fixed as 100. We can observe from Figure 7(d) that the “total computational time increases with the increasing number of P2P nodes too”.

## IX. CONCLUSIONS AND FUTURE WORK

We proposed a robust and efficient blockchain-based access control enabled blockchain solution for PEC in IIoT deployment (PBACS-PECIIoT). We considered private blockchain scenario due to strictly confidential and private data belonging to each IIoT application. The proposed PBACS-PECIIoT is not only secure against various potential attacks, but it also offers various functionality features. The simulation results using the formal security verification under AVISPA automated software tool demonstrate that PBACS-PECIIoT is secure against passive and active attacks. Finally, a detailed comparative study reveals that PBACS-PECIIoT offers better “security features” and more “functionality features”, requires low “communication costs” and comparable “computational costs” as compared to existing relevant recent schemes.

Some future works are as follows. We would like to develop of a real testbed experiment for the whole proposed scheme for implementing the access control and key management parts. Next, we would like to apply “fog computing”, “multi-access edge computing”, and “dew computing” as in

[75] to check if it is possible to come out with an efficient blockchain-based access control technique that can significantly reduce the computation cost.

## ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers and the Associate Editor for their invaluable feedback.

## REFERENCES

- [1] L. D. Xu, W. He, and S. Li, “Internet of Things in Industries: A Survey,” *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.
- [2] “Industrial IoT (IIoT) Market by Device & Technology,” 2020, Accessed on March 2020. [Online]. Available: <https://www.marketsandmarkets.com/Market-Reports/industrial-internet-of-things-market-129733727.html>
- [3] Y. Yang, “A Vision towards Pervasive Edge Computing,” in 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM’19), Miami Beach, FL, USA, 2019, p. 1.
- [4] J. Sengupta, S. Ruj, and S. Das Bit, “A Comprehensive Survey on Attacks, Security Issues and Blockchain Solutions for IoT and IIoT,” *Journal of Network and Computer Applications*, vol. 149, p. 102481, 2020.
- [5] 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.
- [6] C. Zhang, G. Zhou, H. Li, and Y. Cao, “Manufacturing Blockchain of Things for the Configuration of a Data- and Knowledge-Driven Digital Twin Manufacturing Cell,” *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11 884–11 894, 2020.
- [7] C. Zhang, G. Zhou, J. Li, F. Chang, K. Ding, and D. Ma, “A multi-access edge computing enabled framework for the construction of a knowledge-sharing intelligent machine tool swarm in industry 4.0,” *Journal of Manufacturing Systems*, vol. 66, pp. 56–70, 2023.
- [8] M. Castro and B. Liskov, “Practical Byzantine fault tolerance and proactive recovery,” *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 398–461, 2002.
- [9] A. K. Das and B. Bruhadesswar, “An Improved and Effective Secure Password-Based Authentication and Key Agreement Scheme Using Smart Cards for the Telecare Medicine Information System,” *Journal of Medical Systems*, vol. 37, no. 5, p. 9969, 2013.
- [10] S. Chatterjee and A. K. Das, “An effective ECC-based user access control scheme with attribute-based encryption for wireless sensor networks,” *Security and Communication Networks*, vol. 8, no. 9, pp. 1752–1771, 2015.
- [11] D. Mishra, A. K. Das, and S. Mukhopadhyay, “A secure and efficient ECC-based user anonymity-preserving session initiation authentication protocol using smart card,” *Peer-to-Peer Networking and Applications*, vol. 9, no. 1, pp. 171–192, 2016.
- [12] S. Mandal, B. Bera, A. K. Sutrala, A. K. Das, K.-K. R. Choo, and Y. Park, “Certificateless-Signcryption-Based Three-Factor User Access Control Scheme for IoT Environment,” *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3184–3197, 2020.
- [13] M. Wazid, S. Thapliyal, D. P. Singh, A. K. Das, and S. Shetty, “Design and Testbed Experiments of User Authentication and Key Establishment Mechanism for Smart Healthcare Cyber Physical Systems,” *IEEE Transactions on Network Science and Engineering*, 2022.
- [14] B. Bera, A. K. Das, W. Balzano, and C. M. Medaglia, “On the design of biometric-based user authentication protocol in smart city environment,” *Pattern Recognition Letters*, vol. 138, pp. 439–446, 2020.
- [15] A. K. Das, N. R. Paul, and L. Tripathy, “Cryptanalysis and improvement of an access control in user hierarchy based on elliptic curve cryptosystem,” *Information Sciences*, vol. 209, pp. 80–92, 2012.
- [16] S. Zeadally, A. K. Das, and N. Sklavos, “Cryptographic technologies and protocol standards for Internet of Things,” *Internet of Things*, vol. 14, p. 100075, 2021.
- [17] A. K. Das, S. Zeadally, and D. He, “Taxonomy and analysis of security protocols for internet of things,” *Future Generation Computer Systems*, vol. 89, pp. 110–125, 2018.

- [18] S. Roy, S. Chatterjee, A. K. Das, S. Chattopadhyay, N. Kumar, and A. V. Vasilakos, "On the design of provably secure lightweight remote user authentication scheme for mobile cloud computing services," *IEEE Access*, vol. 5, no. 1, pp. 25 808–25 825, 2017.
- [19] S. Banerjee, V. Odelu, A. K. Das, J. Srinivas, N. Kumar, S. Chattopadhyay, and K.-K. R. Choo, "A Provably Secure and Lightweight Anonymous User Authenticated Session Key Exchange Scheme for Internet of Things Deployment," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8739–8752, 2019.
- [20] B. Bera, A. K. Das, and A. K. Sutrala, "Private blockchain-based access control mechanism for unauthorized UAV detection and mitigation in Internet of Drones environment," *Computer Communications*, vol. 166, pp. 91–109, 2021.
- [21] F. Li, J. Hong, and A. A. Omala, "Efficient certificateless access control for industrial Internet of Things," *Future Generation Computer Systems*, vol. 76, pp. 285 – 292, 2017.
- [22] B. M. and K. S. G., "An Authentication Protocol for Future Sensor Networks," *Sensors*, vol. 17, 2017. [Online]. Available: <https://doi.org/10.3390/s17050979>
- [23] J. Xue, C. Xu, and Y. Zhang, "Private Blockchain-Based Secure Access Control for Smart Home Systems," *KSII Transactions on Internet and Information Systems*, vol. 12, no. 12, pp. 6057–6078, 2018.
- [24] X. Li, J. Peng, J. Niu, F. Wu, J. Liao, and K. R. Choo, "A Robust and Energy Efficient Authentication Protocol for Industrial Internet of Things," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1606–1615, 2018.
- [25] M. Luo, Y. Luo, Y. Wan, and Z. Wang, "Secure and efficient access control scheme for wireless sensor networks in the cross-domain context of the IoT," *Security and Communication Networks*, vol. 2018, pp. 1–10, 2018. [Online]. Available: <https://doi.org/10.1155/2018/6140978>
- [26] X. Li, J. Niu, M. Z. A. Bhuiyan, F. Wu, M. Karuppiyah, and S. Kumari, "A Robust ECC-Based Provable Secure Authentication Protocol With Privacy Preserving for Industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3599–3609, 2018.
- [27] X. Zeng, G. Xu, X. Zheng, Y. Xiang, and W. Zhou, "E-AUA: An Efficient Anonymous User Authentication Protocol for Mobile IoT," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1506–1519, 2019.
- [28] S. Mandal, B. Bera, A. K. Sutrala, A. K. Das, K. R. Choo, and Y. Park, "Certificateless Signcryption-Based Three-Factor User Access Control Scheme for IoT Environment," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3184–3197, 2020.
- [29] A. Esfahani, G. Mantas, R. Maticsek, F. B. Saghezchi, J. Rodriguez, A. Bicaku, S. Maksuti, M. G. Tauber, C. Schmittner, and J. Bastos, "A Lightweight Authentication Mechanism for M2M Communications in Industrial IoT Environment," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 288–296, 2019.
- [30] S. Garg, K. Kaur, G. Kaddoum, and K. R. Choo, "Toward Secure and Provable Authentication for Internet of Things: Realizing Industry 4.0," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4598–4606, 2020.
- [31] L. Zhang, Y. Cui, and Y. Mu, "Improving Security and Privacy Attribute Based Data Sharing in Cloud Computing," *IEEE Systems Journal*, vol. 14, no. 1, pp. 387–397, 2020.
- [32] R. Xu, J. Joshi, and P. Krishnamurthy, "An Integrated Privacy Preserving Attribute-Based Access Control Framework Supporting Secure Deduplication," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 2, pp. 706–721, 2021.
- [33] H. Tian, X. Li, H. Quan, C.-C. Chang, and T. Baker, "A Lightweight Attribute-Based Access Control Scheme for Intelligent Transportation System With Full Privacy Protection," *IEEE Sensors Journal*, vol. 21, no. 14, pp. 15 793–15 806, 2021.
- [34] M. Gupta, F. M. Awaysheh, J. Benson, M. Alazab, F. Patwa, and R. Sandhu, "An attribute-based access control for cloud enabled industrial smart vehicles," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4288–4297, 2021.
- [35] Z. Han, X. Li, G. Xu, N. Xiong, E. Merlo, and E. Stroulia, "An effective evolutionary analysis scheme for industrial software access control models," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1024–1034, 2020.
- [36] S. Garg, K. Kaur, G. Kaddoum, and K.-K. R. Choo, "Toward Secure and Provable Authentication for Internet of Things: Realizing Industry 4.0," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4598–4606, 2020.
- [37] M. Amoon, T. Altameem, and A. Altameem, "RRAC: Role based reputed access control method for mitigating malicious impact in intelligent IoT platforms," *Computer Communications*, vol. 151, pp. 238–246, 2020.
- [38] 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.
- [39] Y. Ren, F. Zhu, J. Qi, J. Wang, and A. K. Sangaiah, "Identity Management and Access Control Based on Blockchain under Edge Computing for the Industrial Internet of Things," *Applied Sciences*, vol. 9, no. 10, 2019.
- [40] G. Yu, X. Zha, X. Wang, W. Ni, K. Yu, P. Yu, J. A. Zhang, R. P. Liu, and Y. J. Guo, "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.
- [41] S. Gao, G. Piao, J. Zhu, X. Ma, and J. Ma, "TrustAccess: A Trustworthy Secure Ciphertext-Policy and Attribute Hiding Access Control Scheme Based on Blockchain," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5784–5798, 2020.
- [42] Y. Zhang, M. Yutaka, M. Sasabe, and S. Kasahara, "Attribute-Based Access Control for Smart Cities: A Smart-Contract-Driven Framework," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6372–6384, 2021.
- [43] Y. Nakamura, Y. Zhang, M. Sasabe, and S. Kasahara, "Capability-Based Access Control for the Internet of Things: An Ethereum Blockchain-Based Scheme," in *2019 IEEE Global Communications Conference (GLOBECOM)*, Waikoloa, HI, USA, 2019, pp. 1–6.
- [44] Y. Liu, Q. Lu, S. Chen, Q. Qu, H. O'Connor, K.-K. Raymond Choo, and H. Zhang, "Capability-based IoT access control using blockchain," *Digital Communications and Networks*, vol. 7, no. 4, pp. 463–469, 2021.
- [45] H. Yu, T. Chen, and J. Wang, "A Blockchain-Based Access Control Mechanism for IOT," in *6th International Conference on Electronic Information Technology and Computer Engineering (EITCE '22)*, Xiamen, China, 2023, pp. 25–30.
- [46] A. Vangala, A. K. Das, A. Mitra, S. K. Das, and Y. Park, "Blockchain-Enabled Authenticated Key Agreement Scheme for Mobile Vehicles-Assisted Precision Agricultural IoT Networks," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 904–919, 2023.
- [47] S. M. Awan, M. A. Azad, J. Arshad, U. Waheed, and T. Sharif, "A Blockchain-Inspired Attribute-Based Zero-Trust Access Control Model for IoT," *Information*, vol. 14, no. 2, pp. 1–26, 2023. [Online]. Available: <https://www.mdpi.com/2078-2489/14/2/129>
- [48] R. Canetti and H. Krawczyk, "Universally Composable Notions of Key Exchange and Secure Channels," in *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'02)*, Amsterdam, The Netherlands, 2002, pp. 337–351.
- [49] "MIRACL Cryptographic SDK: Multiprecision Integer and Rational Arithmetic Cryptographic Library," 2020, Accessed on April 2020. [Online]. Available: <https://github.com/miracl/MIRACL>
- [50] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *IEEE Transactions on Computers*, vol. 51, no. 5, pp. 541–552, 2002.
- [51] A. K. Das, "A random key establishment scheme for multi-phase deployment in large-scale distributed sensor networks."
- [52] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.
- [53] H.-F. Huang, "A novel access control protocol for secure sensor networks," *Computer Standards & Interfaces*, vol. 31, pp. 272–276, 2009.
- [54] Y. Zhou, Y. Zhang, and Y. Fang, "Access control in wireless sensor networks," *Ad Hoc Networks*, vol. 5, pp. 3–13, 2007.
- [55] L. Wu, J. Wang, K. R. Choo, and D. He, "Secure Key Agreement and Key Protection for Mobile Device User Authentication," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 319–330, 2019.
- [56] A. K. Das, M. Wazid, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Biometrics-Based Privacy-Preserving User Authentication Scheme for Cloud-Based Industrial Internet of Things Deployment," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4900–4913, 2018.
- [57] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.
- [58] E. Barker, "Recommendation for Key Management," 2016, special Publication 800-57 Part 1 Rev. 4, NIST, 01/2016. Accessed on November 2019. [Online]. Available: <https://nvlpubs.nist.gov/nnistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf>
- [59] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly Secure Key Distribution for Dynamic Conferences," *Information and Computation*, vol. 146, no. 1, pp. 1 – 23, 1998.
- [60] H. Zhang, J. Wang, and Y. Ding, "Blockchain-based decentralized and secure keyless signature scheme for smart grid," *Energy*, vol. 180, pp. 955–967, 2019.

- [61] M. Abdalla, P. A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in 8th International Workshop on Theory and Practice in Public Key Cryptography (PKC'05), Lecture Notes in Computer Science, vol. 3386, Les Diablerets, Switzerland, 2005, pp. 65–84.
- [62] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated Key Exchange Secure against Dictionary Attacks," in Advances in Cryptology —EUROCRYPT 2000, B. Preneel, Ed., Bruges, Belgium, 2000, pp. 139–155.
- [63] L. Wu, J. Wang, K.-K. R. Choo, and D. He, "Secure Key Agreement and Key Protection for Mobile Device User Authentication," IEEE Transactions on Information Forensics and Security, vol. 14, no. 2, pp. 319–330, 2019.
- [64] C. C. Chang and H. D. Le, "A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks," IEEE Transactions on Wireless Communications, vol. 15, no. 1, pp. 357–366, 2016.
- [65] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in Financial Cryptography and Data Security. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 436–454.
- [66] H. Azimy and A. Ghorbani, "Competitive selfish mining," in 2019 17th International Conference on Privacy, Security and Trust (PST), 2019, pp. 1–8.
- [67] M. Davidson and T. Diamond, "On the Profitability of Selfish Mining Against Multiple Difficulty Adjustment Algorithms," Cryptology ePrint Archive, Report 2020/094, 2020, <https://ia.cr/2020/094>.
- [68] L. Vigano, "Automated Security Protocol Analysis With the AVISPA Tool," Electronic Notes in Theoretical Computer Science, vol. 155, pp. 61 – 86, 2006.
- [69] AVISPA, "Automated Validation of Internet Security Protocols and Applications," 2021, <http://www.avispa-project.org/>. Accessed on January 2021.
- [70] ———, "SPAN, the Security Protocol ANimator for AVISPA," 2021, <http://www.avispa-project.org/>. Accessed on January 2021.
- [71] "Raspberry Pi 3 Model B+," 2020, Accessed on April 2020. [Online]. Available: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/>
- [72] "What's the Difference Between a Raspberry Pi and a Computer?" 2022, <https://raspberrytips.com/difference-raspberry-pi-computer/>. Accessed on October 2022.
- [73] D. E. Knuth, The Art of Computer Programming: Seminumerical Algorithms, 3rd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997, vol. 2.
- [74] J. Poon and T. Dryja, "The bitcoin lightning network: Scalable off-chain instant payments," 2016. [Online]. Available: <https://www.bitcoinlightning.com/wp-content/uploads/2018/03/lightning-network-paper.pdf>
- [75] S. Namane and I. Ben Dhaou, "Blockchain-Based Access Control Techniques for IoT Applications," Electronics, vol. 11, no. 14, pp. 1–29, 2022.



**BASUDEB BERA** received "a Ph.D. degree in computer science and engineering from International Institute of Information Technology, Hyderabad, India, in 2022. He also received his M.Sc. degree in mathematics and computing in 2014 from IIT (ISM) Dhanbad, India, and M.Tech. degree in computer science and data processing in 2017 from IIT Kharagpur, India. He is currently a post-doctoral fellow at the National University of Singapore (NUS), Singapore. He was also a post-doctoral fellow at the Singapore University of Technology & Design (SUTD), Singapore. His research interests are cryptography, network security and blockchain technology. He has published over 30 papers in international journals and conferences in his research areas."



**ASHOK KUMAR DAS** (Senior Member, IEEE) "received a Ph.D. degree in computer science and engineering, an M.Tech. degree in computer science and data processing, and an M.Sc. degree in mathematics from IIT Kharagpur, India. He is currently a full Professor with the Center for Security, Theory and Algorithmic Research, IIIT, Hyderabad, India, and also a visiting faculty with the Virginia Modeling, Analysis and Simulation Center, Old Dominion University, Suffolk, VA 23435, USA. His current research interests include cryptography, system and network security including security in smart grid, Internet of Things (IoT), Internet of Drones (IoD), Internet of Vehicles (IoV), Cyber-Physical Systems (CPS) and cloud computing, intrusion detection, blockchain, AI/ML security, and post-quantum cryptography. He has authored over 380 papers in international journals and conferences in the above areas, including over 320 reputed journal papers. He was a recipient of the Institute Silver Medal from IIT Kharagpur. He has been listed in the Web of Science (Clarivate™) Highly Cited Researcher 2022 in recognition of his exceptional research performance. He was/is on the editorial board of IEEE Systems Journal, Journal of Network and Computer Applications (Elsevier), Computer Communications (Elsevier), Journal of Cloud Computing (Springer), Cyber Security and Applications (Elsevier), IET Communications, KSII Transactions on Internet and Information Systems, and International Journal of Internet Technology and Secured Transactions (Inderscience). He also served as one of the Technical Program Committee Chairs of the first International Congress on Blockchain and Applications (BLOCKCHAIN'19), Avila, Spain, June 2019, International Conference on Applied Soft Computing and Communication Networks (ACN'20), October 2020, Chennai, India, and second International Congress on Blockchain and Applications (BLOCKCHAIN'20), L'Aquila, Italy, October 2020. His Google Scholar h-index is 81 and i10-index is 240 with over 18,700 citations."



**SOURAV SAHA** is "currently a Ph.D. student in Computer Science and Engineering at the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad, India. He received his Master of Science (MS) by Research degree in Computer Science and Engineering from Indian Institute of Information Technology, Sri City, Chittoor, India and Bachelor of Technology (B.Tech) in Computer Science and Engineering from Central Institute of Technology, Kokrajhar, India. His research interests include network security and blockchain technology. He has published 10 papers in international journals and conferences in his research areas."



**NEERAJ KUMAR** (Senior Member, IEEE) “received his Ph.D. in CSE from Shri Mata Vaishno Devi University, Katra (J & K), India, and was a postdoctoral research fellow in Coventry University, Coventry, UK. He is working as an Associate Professor in the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology (Deemed to be University), Patiala, India. He has published more than 850 technical research papers in leading journals and conferences. He is on the editorial board of ACM Computing Survey, IEEE Transactions on Sustainable Computing, IEEE Network Magazine, IEEE Communication Magazine, Journal of Networks and Computer Applications (Elsevier), Computer Communications (Elsevier), International Journal of Communication Systems (Wiley), and Security and Privacy (Wiley). His Google Scholar h-index is 111 and i10-index is 572 with over 39,900 citations. He is a senior member of the IEEE.”



**YOUNGHO PARK** (Member, IEEE) received “his BS, MS, and Ph. D degrees in electronic engineering, Kyungpook National University, Daegu, Korea, in 1989, 1991, and 1995, respectively. He is currently a professor at School of Electronics Engineering, Kyungpook National University. In 1996–2008, he was a professor at School of Electronics and Electrical Engineering, Sangju National University, Korea. In 2003–2004, he was a visiting scholar at School of Electrical Engineering and Computer Science, Oregon State University, USA. His research interests include computer networks, multimedia, and information security.”

• • •



**SK HAFIZUL ISLAM** (Senior Member, IEEE) received “the M.Sc. degree in applied mathematics from Vidyasagar University, Midnapore, India, in 2006, and the M.Tech. degree in Computer Application and the Ph.D. degree in Computer Science and Engineering in 2009 and 2013, respectively, from Indian Institute of Technology [IIT (ISM)] Dhanbad, Jharkhand, India, under the INSPIRE Fellowship Ph.D. Program (funded by the Department of Science and Technology, Government of India). He is currently an Assistant Professor in the Department of Computer Science and Engineering, Indian Institute of Information Technology Kalyani (IIIT Kalyani), West Bengal, India. Before joining the IIIT Kalyani, he was an Assistant Professor with the Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani (BITS Pilani), Rajasthan, India. He has more than ten years of teaching and thirteen years of research experience. He has authored or co-authored 130 research papers in journals and conference proceedings of international repute. His research interests include cryptography, information security, neural cryptography, lattice-based cryptography, IoT & blockchain security, and deep learning. He has edited four books for the publishers Scrivener-Wiley, Elsevier, and CRC Press. He is an Associate Editor for IEEE Transactions on Intelligent Transportation Systems, IEEE Access, International Journal of Communication Systems (Wiley), Telecommunication Systems (Springer), IET Wireless Sensor Systems, Security and Privacy (Wiley), Array-Journal (Elsevier), Wireless Communications and Mobile Computing (Hindawi), and Journal of Cloud Computing (Springer). He received the University Gold Medal, the S. D. Singha Memorial Endowment Gold Medal, and the Sabitri Parya Memorial Endowment Gold Medal from Vidyasagar University in 2006. He also received the University Gold Medal from IIT(ISM) Dhanbad in 2009 and the OPERA award from BITS Pilani in 2015. He is a senior member of IEEE and a member of ACM.”