Designing Blockchain-Based Access Control Protocol in IoT-Enabled Smart-Grid System

Basudeb Bera, *Student Member, IEEE*, Sourav Saha, *Student Member, IEEE*, Ashok Kumar Das, *Senior Member, IEEE*, and Athanasios V. Vasilakos, *Senior Member, IEEE*

***Abstract*—We design a new blockchain-based access control protocol in IoT-enabled smart-grid system, called DBACP-IoTSG. Through the proposed DBACP-IoTSG, the data is securely brought to the service providers from their respective smart meters (SMs). The peer-to-peer (P2P) network is formed by the participating service providers, where the peer nodes are responsible for creating the blocks from the gathered data securely from their corresponding SMs and adding them into the blockchain after validation of the blocks using the voting-based consensus algorithm. In our work, the blockchain is considered as private because the data collected from the consumers of the SMs are private and confidential. By the formal security analysis under the random oracle model, nonmathematical security analysis and software-based formal security verification, DBACP-IoTSG is shown to be resistant against various attacks. We carry out the experimental results of various cryptographic primitives that are needed for comparative analysis using the widely used multiprecision integer and rational arithmetic cryptographic library (MIRACL). A detailed comparative study reveals that DBACP-IoTSG supports more functionality features and provides better security apart from its low communication and computation costs as compared to recently proposed relevant schemes. In addition, the blockchain implementation of DBACP-IoTSG has been performed to measure computational time needed for the varied number of blocks addition and also the varied number of transactions per block in the blockchain.**

***Index Terms*—Access control, blockchain, Internet of Things (IoT), security, smart grid.**

I. Introduction

DUE TO rapid population growth and advancement of Internet-of-Things (IoT) systems the demand and popularity of the IoT-based smart grid is growing rapidly. The IoT-based smart grid is embedded with smart sensors, objects, and actuators (IoT smart devices). It provide a reliable transmission of energy which makes the system automated and efficient, and also improves the economic benefits and deliver quality of services [1]. Smart grid is one of the foundations of the cyber–physical systems (CPSs). In CPS, the sensors, smart objects or physical devices are integrated with an interconnected network that helps in delivering the services efficiently [2]. The communication in a smart grid is via a bi-directional way. IoT-based smart grid system has numerous advantages, but at the same time it has also numerous challenges, such as centralized control, transparency, poor interoperability, and privacy and security issues, including energy trading between untrusted/nontransparent networks [3], auditing and verifying of transaction records in the system [4]. Such limitations in the conventional centralized system needs to be addressed using the decentralized approach. Also, a good amount of efforts is needed to improve "transparency," "efficiency," "reliability," "resilience," "fraud prevention" as well as "quality of services" in a smart grid system [5], [6]. The blockchain-based smart grid system is one of the emerging technology that can give the solution for present "centralized infrastructures." Various intrinsic applications have comeup recently in the blockchain of Things (BCoTs) [7]. Some potential applications include: 1) "smart grid" [8], [9]; 2) "healthcare system" [10], [11]; 3) "Internet of Vehicles (IoV)" [12]; 4) "smart manufacturing" [5]; 5) "IoT applications" [13]–[17]; 6) "Industry 4.0" [18]; and 7) "smart transportation system" [19].

There are several advantages of using the "blockchain-based smart grid infrastructure," because "decentralization," "immutability," "transparency," "confidentiality," and "trust" are maintained [20]. Transparency property means that "when an entity's real identity is made secure, one can still view all the transactions that were done by their public addresses." Immutability property allows that "once a block having the information is inserted into the private/public blockchain, it can not be tampered later." Blockchain is of three categories: 1) "public blockchain"; 2) "private blockchain"; and 3) "consortium blockchain." The public blockckchain is "open to anybody to join, access, send, verify and receive transactions of the blocks in the network." Some applications that use public blockchain are "cryptocurrency (Bitcoin)" [21] and "Ethereum." In a private blockckchain, it is a considered as a fully trusted network. In this case, "the access is only granted to a particular entity or a group of trusted

Manuscript received July 17, 2020; revised September 13, 2020; accepted October 8, 2020. Date of publication October 13, 2020; date of current version March 24, 2021. This work was supported in part by the Ripple Centre of Excellence Scheme, CoE in Blockchain (Sanction No. IIIT/Research and Development Office/Internal Projects/001/2019), IIIT Hyderabad, India, and in part by the Mathematical Research Impact Centric Support (MATRICS) project funded by the Science and Engineering Research Board, India, under Grant MTR/2019/000699. *(Corresponding author: Ashok Kumar Das.)*

Basudeb Bera, Sourav Saha, and Ashok Kumar Das are with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology Hyderabad, Hyderabad 500032, India (e-mail: basudeb.bera@research.iiit.ac.in; sourav.saha@research.iiit.ac.in; iitkgp.akdas@gmail.com).

Athanasios V. Vasilakos is with the School of Electrical and Data Engineering, University Technology Sydney, Sydney, NSW 2007, Australia, also with the Department of Computer Science and Technology, Fuzhou University, Fuzhou 350116, China, and also with the Department of Computer Science, Electrical and Space Engineering, Lulea University of Technology, 97187 Lulea, Sweden (e-mail: th.vasilakos@gmail.com).

Digital Object Identifier 10.1109/JIOT.2020.3030308

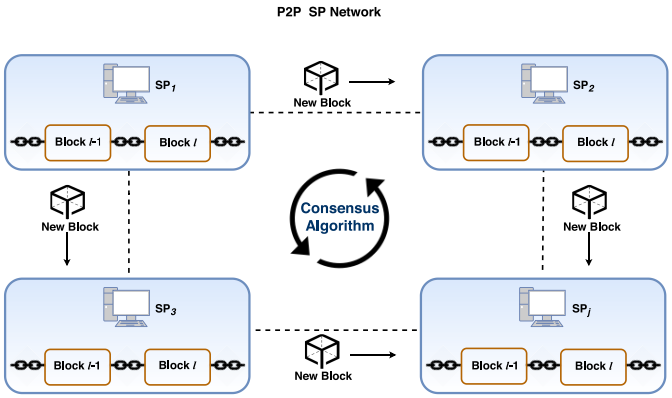

Fig. 1. Blockchain-based P2P network of SPs (SP_j) in smart grid (adapted from [9]).

entities" and also "the owner of the network mainly decides which entity will perform a specific task." A consortium blockchain is treated as a "combination of both public and private blockchains." Consensus algorithms are needed in order to achieve "consensus among the nodes involved in a peer-to-peer (P2P) blockchain network." Some broadly accepted consensus algorithms are "Byzantine fault tolerance (BFT)," "practical BFT (PBFT)," "Proof-of-Work (PoW)," and "Proof-of-Stake (PoS)" [5].

Blockchain is a distributed ledger and it validates the transactions without any intervention of a trusted third party (TTP). Danzi *et al.* [22] designed a scheme which was implemented by the nodes containing in a blockchain network. Their scheme aggregates the blockchain data in periodic updates. Thus, it helps in reducing the communication overheads of the IoT devices connected in the blockchain network. Ai *et al.* [23] pointed out that there is a need for reliable data diffusion for the IoT deployment. Moreover, they observed that the random placement makes the limited resources as well as unattended characteristics of the existing IoT smart devices. Therefore, in such a scenario the wireless communication can be easily exposed to malicious users, which leads to a vulnerable area for different malicious threats, including Sybil attack and also wormhole attack. To cope with these limitations, they proposed a "smart collaborative routing protocol." In this scheme, with the help of the game theoretic approach, the inspecting node is elected in order to monitor the behaviors of IoT network. In addition, Song *et al.* [24] also proposed a "smart collaborative tracking scheme" by probing the advanced "parameter prediction skills" and "improved particle filter" mechanisms. It was observed that their scheme can improve tracking accuracy along with it can also restrain particle dilution.

A blockchain-based smart grid system contains several entities, such as trusted registration authority (RA), service providers (SPs), IoT-enabled SMs, and users associated with an SM. Fig. 1 shows the role of P2P network of SPs in a smart grid. SPs organize the electricity allocation and energy trading system, and SMs are responsible for monitoring the power utilization and they maintain the pricing to the consumers (users). SMs can be deployed in the homes, and an attacker may capture some SMs and use the secure data stored into it [25]. The communications between SPs and SMs must be secure so that passive/active attacks should be not possible [26]. To ensure the security and privacy of the users' private information, it is extremely required to design a secure and efficient access control scheme between SMs and SPs. With the help from the blockchain technology, the secure data can be stored in the form of blocks in a private blockchain. The SPs involved in the P2P SP network are responsible in validating the new blocks before adding them into the blockchain using the consensus algorithm. To mitigate these issue, we aim to design a blockchain-based access control mechanism in an IoT-enabled smart grid environment.

A. Motivation

Nowadays, modern power systems suffer from various challenges, for instance, ever-expanding electrical energy demand, exponential growth in renewable energy sector, adaptation of large-scale IoT devices and also several security threats posed in CPS. Another goal of IoT-enabled smart grid system is to maintain the reliability and stability of the system [20]. These challenges put in discovering the security solutions for reliable operations of the power system.

In recent years, there are several blockchain-based potential applications in smart grid domain [27]. Some of the smart grid applications using blockchain are given as follows.

1) *Power Generation:* With the help of the blockchain, the dispatching organizations can have a full knowledge about the entire operation condition of a power grid in a real-time viewpoint. This helps the organizations to develop dispatching actions in order to maximize profits.
2) *Power Transmission and Distribution:* The automation and control centers become the decentralized systems with the help of the blockchain technology that conquer the main challenges faced in the "'traditional centralized systems."
3) *Power Consumptions:* Using the blockchain, it can manage the energy trading among the prosumers and various energy storage systems (e.g., electric vehicles)

Recently, blockchain-based solution is one of the promising technology that can be used for providing the security in the smart grid environment because of its uniqueness and also decentralized architecture. Since the communication among the users, SMs and SPs take place via open environment, an adversary has opportunity to tamper with the data, and can perform various prospective attacks, including "replay," "impersonation," "man-in-the-middle" and "ephemeral secret leakage (ESL)" attacks. Apart from these attacks, the adversary can also trace the communicated messages and therefore, anonymity and untraceability are the two main important functionality attributes that an access control security protocol should support those features. To deal with these issues, we proposed a new decentralized blockchain-based access control protocol in IoT-enabled smart-grid system (DBACP-IoTSG), in which the data is collected from the SMs securely by their respective SPs before forming the blocks and adding those blocks in the blockchain through a voting-based consensus mechanism in the P2P SP network. Because the blockchain provides transparency and immutability properties, once the block is added in the blockchain it can not be tampered by any

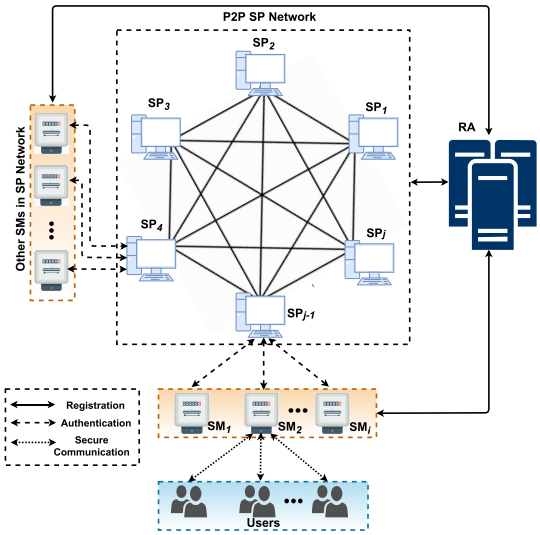

Fig. 2. Blockchain-based IoT-enabled smart grid architecture without TTP (adapted from [9]).

adversary or even by legal entities in the smart grid network, and anybody can see the information stored in the block. In this work, we mainly concentrate on private blockchain because the collected data from the SMs by the SPs are confidential as well as private.

B. Research Contributions

The main contributions in this work are listed as follows.

1) We propose a novel DBACP-IoTSG, based on the network model provided in Fig. 2. In DBACP-IoTSG, registration of SMs and SPs is performed in offline mode prior to deployment in the IoT-enabled smart-grid environment. The access control phase associated with DBACP-IoTSG permits a smart meter SM_i to mutually authenticate with its associated service provider SP_j through "node authentication" process and then to establish a session key among them for secret communication through "key establishment" process. The pairwise secret keys among the SPs are used for secure consensus procedure. DBACP-IoTSG also supports the dynamic node addition mechanism.
2) We then provide a detailed mechanism for a new block creation and addition in the blockchain through the PBFT-based consensus mechanism [28].
3) The proposed DBACP-IoTSG allows secure leader selection in the P2P SP network that is responsible for a block verification and addition in the blockchain using the voting-based PBFT consensus algorithm.
4) The proposed DBACP-IoTSG allows to preserve both anonymity and untraceability properties that are extremely needed in an IoT-enabled smart grid environment. In addition, DBACP-IoTSG also permits dynamic SM addition phase after initial deployment in an event that if some service providers SP_j may become faulty nodes or some smart meters SM_i may be physically compromised by an adversary. Furthermore, DBACP-IoTSG resists a crucial active attack under the present *de facto* model, known as the Canetti and Krawczyk's model (CK-adversary model) [29] (see the threat model in Section III-B), known as "ESL" attack.
5) Next, we apply the threat model given in Section III-B to provide a rigorous security analysis through the formal security under the broadly accepted "real-or-random (ROR) oracle model" [29], informal (nonmathematical) security analysis and also formal security verification using the widely used automated validation of Internet security protocols and applications (AVISPA) automated software tool [30] through simulation.
6) We also provide the experimental results of various cryptographic primitives that are needed for comparative analysis using the widely used "multiprecision integer and rational arithmetic cryptographic library (MIRACL)" [31].
7) A detailed comparative analysis among DBACP-IoTSG and other relevant protocols in smart grid environment shows that DBACP-IoTSG supports better security and provides more functionality attributes, and also requires less communication and computation costs.
8) Finally, the blockchain implementation of DBACP-IoTSG has been carried out in order to measure computational time required for the varied number of blocks addition as well as the varied number of transactions per block in the blockchain.

C. Article Outline

This article outline is as follows. Section II discusses the related work in the IoT-enabled smart grid environment for various access control and authentication mechanisms. The network and threat models are described in Section III. Section IV proposes a novel DBACP-IoTSG. While Section V provides a rigorous security analysis on DBACP-IoTSG to resist various attacks, Section VI provides the experiments of various cryptographic primitives using the "MIRACL" [31]. Section VII gives a detailed comparative analysis among DBACP-IoTSG and other related schemes. The blockchain implementation of the proposed DBACP-IoTSG has been demonstrated in Section VIII. The concluding remarks are then provided in Section IX.

II. Related Work

Musleh *et al.* [20] presented a survey work by considering several aspects, mechanisms, advantages, and also research challenges if the blockchain-based solution is applied in the smart-grid environment. Furthermore, they presented frameworks that are essentials for blockchain-based applications for smart grid. They also emphasized that the blockchain is appropriate for utilizing the cyber–physical layer of the smart grid. They also mentioned that the power grid will turn out to be a splendid usage of blockchain technology like other applications, such as industrial sectors.

Andoni *et al.* [6] examined several industrial and academic sources to present basics of blockchain related technologies, such as "system architectures" and "distributed consensus algorithms," crucial aspects of performance for blockchain

related ecosystems. They particularly showcased some specific domains in which innovation is essential for energy system stakeholders as well as industrial entities.

Kim and Huh [32] presented a smart grid design which replies on power trading system and blockchain concept. Their design supports an architecture that can be applied for stable P2P transactions for transitional smart contact mechanisms in order to stably to trade power information of an already existing smart grid system.

Wang *et al.* [33] examined various related issues in smart grid system, such as need for "removal of the centralized control and transition to a distributed architecture," privacy and security aspects, auction pricing approach at the settlement time, minimization of cost and maximization of benefit of the system.

Aitzhan and Svetinovic [4] discussed the issue of supporting the transaction security in a decentralized smart grid energy trading system, where it does not rely on a TTP. Moreover, through implementation on a "token-based private decentralized energy trading system," they showed that such a system allows the peer entities to negotiate trading prices in an anonymous way and also to execute trading transactions in secure manner. To provide security and privacy at a certain level, they applied multisignatures, blockchain and anonymous "encrypted message propagation streams."

In recent years, several authentication and access control schemes are designed in the smart grid systems [26], [34]–[44]. An authentication protocol designed by Nicanfar *et al.* [39] was applicable for a home network, where an SM can authenticate mutually with an authorized server. Li *et al.*'s mutual authentication scheme [38] designed for smart grid system provides secure communication and it relies on the Merkle hash tree concept. Another authentication scheme designed by Chan and Zhou [34] provided a "two-factor cyber–physical device authentication" in order to provide security in a smart grid system.

Qi and Chen [43] proposed an efficient authenticated key agreement mechanism for smart grid environment using the "elliptic curve Qu-Vanstone (ECQV) implicit certificate" as the building block. In their scheme, a mutual authentication takes place between an SM and an SP and a session key is established at the end of the mutual authentication. However, their scheme does not support the blockchain solution. Chaudhry *et al.* [44] proposed an authentication mechanism for "demand response management" (DRMAS) in a smart grid edge computing-based environment. In this scheme, the blockchain is not also supported.

Later, an anonymous key distribution method was suggested by Tsai and Lo [26]. Their method relies on "bilinear pairings" and elliptic curve cryptography (ECC), which supports mutual authentication among an SM and an SP. Also, a session key among them is crated for secret communication once mutual authentication is successful. Unfortunately, their scheme is not resilient to ESL attack, and also it does not offer "strong credentials' privacy of an SM" [40]. To eradicate the limitations of Tsai and Lo's scheme [26], another authentication mechanism was suggested in [40]. He *et al.* [36] also designed another "ECC-based anonymous key distribution scheme" for a smart grid environment to reduce communication as well as computation overheads as compared to Tsai and Lo's protocol [26].

An authentication mechanism designed by Mahmood *et al.* [45] fails to support "anonymity" feature because they transmitted the SM's actual identity openly in the network. In addition, their scheme fails to prorect "known session-specific temporary information attack (ESL attack)," and also does not provide "perfect forward security," and "private keys leakage security" [46]. Furthermore, Mahmood *et al.* [47] also suggested another authentication mechanism for smart grid system that is based on edge computing paradigm. Unfortunately, Wang *et al.* [48] indicated that mutual authentication is not achieved in the scheme [47] since the validity of utility control is not verified by an SM. In addition, Wang *et al.* [48] designed a mutual authentication mechanism that utilizes blockchain technology. Because of utilization of blockchain, their protocol provides conditional privacy issue and also key management.

Gai *et al.* [49] developed a model for "permissioned blockchain edge" in a smart grid system. Their designed model is able to provide privacy protection along with energy security by applying both blockchain technology and edge computing facility. They further applied group signatures and channel authorization mechanisms to assure the validity of the involved users in the smart grid.

Zhang *et al.* [9] designed a "decentralized keyless signature scheme based on a consortium blockchain" to develop an effective key management. In their approach, the SMs send requests and then receive the responses using a P2P blockchain network for data transmission. They suggested a decentralized consensus mechanism that does not need a TTP or a trusted anchor. Zhou *et al.* [42] proposed an access control mechanism using blockchain in the power system, which relies on "identity-based combined encryption," "digital signature" and "signcryption." Their approach solves the "key escrow" issue of the distrustful third parties.

Recently, Yao *et al.* [50] designed a "decentralized autonomous organization (DAO) trading platform" for an industrial IoT environment that relies on blockchain network assisted with cloud mining concept. They further modeled the "computational resource management and pricing problem" along the miners and the resource provider as a "Stackelberg game." In addition, they applied a "multiagent reinforcement learning" technique in order to attain the near-optimal strategy.

Table I summarizes various existing competing authenticated key agreement schemes and the proposed scheme (DBACP-IoTSG) with respect to their cryptographic techniques used, advantages and limitations/drawbacks.

III. System Models

In this section, we follow the network and threat models that are applied in describing and also in analyzing the proposed scheme (DBACP-IoTSG).

A. Network Model

The network model considered in this work is shown in Fig. 2. In this model, several users are associated with a smart

TABLE I
CRYPTOGRAPHIC TECHNIQUES, ADVANTAGES, AND LIMITATIONS OF EXISTING AUTHENTICATION/ACCESS CONTROL SCHEMES IN SMART GRIDS

Scheme	Year	Cryptographic Techniques	Advantages	Drawbacks/Limitations
Tsai and Lo [26]	2016	* ECC * Bilinear pairings * Hash functions * Modular exponentiations	* Mutual authentication * Session key establishment	* Vulnerable to ephemeral secret leakage (ESL) attack * No strong credentials' privacy of smart meter * High computational cost * Does not support blockchain solution
Mahmood *et al.* [45]	2018	* One-way hash functions * ECC	* Mutual authentication * Session key establishment	* Vulnerable to ESL attack * No anonymity * No perfect forward security * No private keys leakage security * Does not support blockchain solution
Mahmood *et al.* [47]	2018	* Bilinear pairing * ECC * One-way hash functions * Modular exponentiations	* Key agreement	* No mutual authentication * Does not support blockchain solution * Vulnerable to ESL attack * High computational cost
Wang *et al.* [48]	2019	* One-way hash functions * ECC	* Mutual authentication * Key agreement * Support blockchain solution	* No voting-based consensus mechanism for block mining in blockchain * No secure leader selection in P2P network
Zhou *et al.* [42]	2019	* Bilinear pairings * ECC * One-way hash functions	* Support blockchain solution * Signcryption-based access control	* No secure leader selection in P2P network * Vulnerable to ESL attack * No perfect forward security * No dynamic nodes addition after initial deployment * High computational cost
Qi and Chen [43]	2020	* ECC * One-way hash functions	* Mutual authentication * Session key agreement	* Does not support blockchain solution * No dynamic nodes addition after initial deployment
Chaudhry *et al.* [44]	2020	* ECC * One-way hash functions	* Mutual authentication * Session key agreement	* Does not support blockchain solution
Proposed (DBACP-IoTSG)	2020	* ECC * One-way hash functions	* Mutual authentication * Session key agreement * Support blockchain solution	* Needs to implement in real-world environment as future research work

meter SM_i and a group of SMs are also associated with a service provider SP_j. A group of SPs will form a P2P SP network, which is called as the P2P SP network. There is a trusted RA which is responsible for registering all the installed smart meters SM_i and service providers SP_j in offline mode. The *RA* performs the registration process securely.

The communication between the users and a smart meter SM_i takes place via secure communication, whereas a smart meter SM_i and a service provider SP_j communicate securely using a session key established among them with the help of an access control mechanism. In addition, the SPs in the SP network also establish secret pairwise keys among them for their secure communications. Under this network model, SM_i first gathers the data secretly from its associated users and then the collected data is brought secretly to the service provider SP_j under which the smart meters SM_i are regsitered with SP_j. SP_j then forms transactions using the collected data and creates a block. Next, the new created block can be added in the existing blockchain provided that the consensus among the SPs in the SP network is performed. Once a block is added in the blockchain, the modification or deletion of that block is not permitted to maintain "immutability" property.

B. Threat Model

For our proposed DBACP-IoTSG, we contemplate the widely accepted "Dolev–Yao (DY) threat model" [51]. According to the DY model, an adversary $\mathcal{A}$ can insert malicious information, modify or delete the message contents, apart from intercepting the messages among the communicated entities in the IoT-enabled smart grid environment. Furthermore, the end-point communicating entities (users, SMs and SPs) are not contemplated as trustworthy parties in the network. We also assume that some SMs may be physically captured by $\mathcal{A}$ because the SMs can not be monitored in 24×7. Once an SM is compromised physically, all the stored credentials in its memory can be extracted by $\mathcal{A}$ with the help of advanced power analysis attack [52]. In addition, we adopt the recently contemplated *de facto* model, known as the Canetti and Krawczyk's model (CK-adversary model) [29] in the proposed DBACP-IoTSG. Under the CK-adversary model, $\mathcal{A}$ not only can intercept the messages as in the DY model, but can also compromise secret credentials, secret keys and even session states if those information are available in insecure memory of the devices (SM_i and SP_j) during the access control phase [29].

IV. PROPOSED SCHEME

This section proposes a new DBACP-IoTSG, based on the architecture shown in Fig. 2. DBACP-IoTSG contains several phases, namely, 1) system setup; 2) registration of SMs and SPs; 3) access control; 4) key management among SPs; 5) block formation and addition in the blockchain; and 6) new SMs addition after initial deployment in the smart grid environment.

To protect replay attack, we apply both random numbers and current timestamps generated by the entities in the network. It is thus assumed that the entities in the network are synchronized with their clocks. It is also a typical assumption applied in many recent authentication and access control protocols in IoT deployment [1], [53]–[57]. We use various notations tabulated in Table II for describing and also analyzing the proposed DBACP-IoTSG.

TABLE II
NOTATIONS AND THEIR SIGNIFICANCE

Symbol	Significance
$E_q(a,b)$	A non-singular elliptic curve of the form: "$y^2 = x^3 + ax + b \pmod q$ with $4a^3 + 27b^2 \neq 0 \pmod q$"
G	A base point in $E_q(a,b)$ whose order is n as big as q
$k.G$	Elliptic curve point multiplication; $k.G = G + G + \cdots + G\,(k\,times)$
$Q + R$	Elliptic curve point addition; $Q, R \in E_q(a,b)$
$u * v$	Ordinary modular multiplication in $GF(q)$
RA, ID_{RA}	Trusted registration authority and its real identity
RID_{RA}	Pseudo-identity of the RA
mk_{RA}, Pub_{RA}	Private and public keys of RA, respectively, $Pub_{RA} = mk_{RA}.G$
SP_j	j^{th} service provider
SPN	P2P SP network of all registered service providers
ID_{SP_j}, RID_{SP_j}	Real and pseudo-identities of SP_j, respectively
TID_{SP_j}	Temporary identity of SP_j
k_{SP_j}, Pub_{SP_j}	Private and public keys of SP_j, respectively, $Pub_{SP_j} = k_{SP_j}.G$
$f(x,y)$	A symmetric bivariate t-degree polynomial over the Galois field $GF(q)$: $f(x,y) = \sum_{i=0}^{t}\sum_{j=0}^{t} a_{ij}x^i y^j$ where $a_{ij} \in Z_q = \{0,1,2,\cdots,q-1\}$
SM_i, ID_{SM_i}	i^{th} smart meter and its real identity
TID_{SM_i}, RID_{SM_i}	Temporary and pseudo identities of SM_i, respectively
TC_{SP_j}, TC_{SM_i}	Temporal credentials of SP_j and SM_i, respectively
$Cert_{SP_j}, Cert_{SM_i}$	Certificates issued by the RA to SP_j and SM_i, respectively
RTS_{SP_j}, RTS_{SM_i}	Registration timestamps of SP_j and SM_i, respectively
$\|\|, \oplus$	Concatenation & bitwise XOR operations, respectively
TS_x	"Current timestamp generated by an entity X" (i.e., SP_j or SM_i)
ΔT	"Maximum transmission delay associated with a message"
$h(\cdot)$	"Collision-resistant cryptographic one-way hash function"
$EC(\cdot)/DC(\cdot)$	Symmetric encryption/decryption
$EP(\cdot)/DP(\cdot)$	"Public key encryption/decryption"
MAC	"Message authentication code"

The entities involved in DBACP-IoTSG include the trusted RA, the smart meters SM_i $(i = 1, 2, \ldots, n_{sm})$ and the service providers SP_j $(j = 1, 2, \ldots, n_{sp})$, where n_{sm} and n_{sp} denote the number of SMs and SPs to be deployed initially in the network. All the involved service providers SP_j form a P2P blockchain network, called SPN, which are responsible for creating blocks for transactions that are securely received from their respective smart meters SM_i. A leader from the SPN is selected, which is responsible for adding the block after running the consensus algorithm.

In this article, we consider an access control mechanism which has basically the following two tasks [58].

1) *Node Authentication:* This task demands that the newly joined nodes (SM_i and SP_j) must authenticate themselves to with other nodes to prove that they are authorized registered nodes to access the services from each other.
2) *Key Establishment:* This task requires that a newly deployed node needs to establish the shared secret pairwise key with its neighbor nodes after the mutual authentication between them is done satisfactorily. The established secret keys are then used by the nodes for secret communication.

The detailed description of each phase is given as follows.

A. System Initialization Phase

The trusted *RA* is responsible for picking the system parameters using the following steps.

S1: The *RA* selects a "nonsingular elliptic curve of the form: $y^2 = x^3 + ax + b \pmod q$ over the Galois field $GF(q)$, where q is a large prime and $4a^3 + 27b^2 \neq 0 \pmod q$ with $\mathcal{O}$ as the point at infinity or zero point." The *RA* also picks a base point $G \in E_q(a,b)$ whose order will be as big as q, say n, that is, $n.G = \mathcal{O}$.

S2: The *RA* selects an identity ID_{RA}, and picks a master key mk_{RA} as the private key and its respective public key as $\text{Pub}_{RA} = mk_{RA}.G$, and also its pseudoidentity $RID_{RA} = h(\text{ID}_{RA} \| mk_{RA})$.

S3: The *RA* then picks a "one-way cryptographic hash function $h : \{0,1\}^* \rightarrow \{0,1\}^{l_h}$ which takes an arbitrary length input string $x \in \{0,1\}^*$ and produces a fixed length output string of l_h bits, $h(x) \in \{0,1\}^{l_h}$." For instance, $h(\cdot)$ can be taken as "Secure Hash Algorithm (SHA-1) which produces 160-bit hash value and for more security, it can be SHA-256 or SHA-512 [59]." Moreover, to sign a message, the *RA* selects the "elliptic curve digital signature algorithm (ECDSA)" [60] which contains the signature generation and verification algorithms.

S4: Finally, the *RA* keeps mk_{RA} as its private key, and publishes other parameters $\{E_q(a,b),\ h(\cdot),\ G,\ \text{Pub}_{RA}\}$ which are publicly accessible to all the entities in the network.

B. Registration Phase

This phase is executed by the *RA* in offline mode for the purpose of registering all the deployed smart meters SM_i, $(i = 1, 2, \ldots, n_{sm})$ and also the service providers SP_j, $(j = 1, 2, \ldots, n_{sp})$. It is worth noticing that the registered service providers SP_j will be part of the SPN.

1) Smart Meter Registration Phase: The following steps are essential to complete the registration process of each deployed SM_i.

RSM1: For each SM_i, the *RA* picks a unique real identity ID_{SM_i}, a random temporary identity TID_{SM_i} and calculates pseudorandom identity $RID_{SM_i} = h(\text{ID}_{SM_i} \| mk_{RA} \| RTS_{SM_i})$, where the registration timestamp of SM_i is RTS_{SM_i}.

RSM2: For each SM_i, the *RA* picks a random private key k_{SM_i} and calculates its respective public key $\text{Pub}_{SM_i} = k_{SM_i}.G$. In addition, the *RA* calculates the temporal secret for each SM_i as $TC_{SM_i} = h(\text{ID}_{SM_i} \| mk_{RA} \| k_{SM_i} \| RTS_{SM_i})$ and certificate as $\text{Cert}_{SM_i} = k_{SM_i} + h(RID_{RA} \| \text{Pub}_{RA} \| RID_{SM_i}) * mk_{RA} \pmod q$ using its own private key mk_{RA}. The *RA* picks a random private key br_{SM_i} and calculates its respective public key $\text{BPub}_{SM_i} = br_{SM_i}.G$ for each SM_i.

RSM3: Finally, the *RA* loads the credentials $\{TID_{SM_i}, RID_{SM_i}, TC_{SM_i}, RID_{RA}, \text{Cert}_{SM_i}, (br_{SM_i}, \text{BPub}_{SM_i})\}$ prior to its placement in the network, and declares all public keys Pub_{SM_i} as public. It is also worth noting that all the information $\{TID_{SM_i}, RID_{SM_i}, TC_{SM_i}, \text{Cert}_{SM_i}, br_{SM_i}\}$ are distinct for each deployed SM_i through the network. The *RA* deletes all the generated private keys k_{SM_i} and br_{SM_i} for each registered SM_i.

2) Service Provider Registration Phase: Similar to the SM registration process, the *RA* proceeds to execute the following steps to complete the registration of each deployed service provider SP_j under which the smart meters SM_i, $(i = 1, 2, \ldots, n_{sm})$ will be functional.

RSP1: For each SP_j, the *RA* first picks a unique real identity ID_{SP_j}, a random temporary identity TID_{SP_j} and computes its pseudoidentity as $RID_{SP_j} = h(\text{ID}_{SP_j} \| mk_{RA} \| RTS_{SP_j})$, where the registration timestamp of SP_j is denoted by RTS_{SP_j}.

RSP2: For each registered SP_j, the *RA* also picks a random private key k_{SP_j} and calculates its respective public key

$\text{Pub}_{SP_j} = k_{SP_j}.G$, and also the temporal secret as $TC_{SP_j} = h(\text{ID}_{SP_j} \| mk_{RA} \| k_{SP_j} \| RTS_{SP_j})$ and certificate using its own private key mk_{RA} as $\text{Cert}_{SP_j} = k_{SP_j} + h(RID_{RA} \| \text{Pub}_{SP_j}) * mk_{RA} \pmod q$.

RSP3: For establishing pairwise secret keys among the SPs (see Section IV-D), we apply the "polynomial-based key distribution approach" as suggested by Blundo *et al.* [61]. To achieve this goal, the *RA* generates a "t-degree bivariate symmetric polynomial of the form $f(x, y) = \sum_{i=0}^{t} \sum_{j=0}^{t} a_{ij} x^i y^j$ over $GF(q)$," where the coefficients $a_{ij} \in Z_q = \{0, 1, 2, \ldots, q-1\}$, with the property that $f(x, y) = f(y, x)$, and calculates a polynomial share for SP_j as $f(RID_{SP_j}, y) = \sum_{i=0}^{t} \sum_{j=0}^{t} a_{ij} RID_{SP_j}^i y^j \pmod q$, which turns out to be a t-degree univariate polynomial. Note that to store this polynomial share, SP_j needs to store $(t+1)$ coefficients, which is equivalent to have storage cost of $(t+1)\log_2(q)$ bits as each coefficient is from $GF(q)$.

RSP4: Finally, the *RA* stores the credentials $\{(TID_{SP_j}, RID_{SP_j}), TC_{SP_j}, RID_{RA}, \text{Cert}_{SP_j}, f(RID_{SP_j}, y), \{(TID_{SM_i}, RID_{SM_i}) | i = 1, 2, \ldots, n_{sm}\}\}$ in SP_j, deletes all the generated private keys k_{SP_j} of the registered service providers SP_j, and declares all public keys Pub_{SP_j} as public. In addition, the *RA* also stores $\{(TID_{SP_l}, RID_{SP_l}) | l \neq j, j = 1, 2, \ldots, n_{sp}\}$ in SP_j corresponding to all other service providers SP_l.

C. Access Control Phase

Through this phase, a smart meter SM_i will be able to authenticate with its respective service provider SP_j with the help of *node authentication* task, and then after mutual authentication between them, they will generate a common session key with the help of *key establishment* task. The detailed discussion is given through the following steps.

AC1: SM_i being the initiator node generates a random secret $r_1 \in Z_q^*$ and current timestamp TS_1, and then calculates $R_1 = h(r_1 \| RID_{SM_i}).G$, $X_1 = h(TC_{SM_i} \| TS_1) \oplus h(RID_{RA} \| RID_{SM_i} \| TS_1)$ and $X_2 = h(TID_{SM_i} \| RID_{SM_i} \| RID_{RA} \| R_1 \| \text{Cert}_{SM_i} \| TS_1)$. Next, SM_i sends the "node authentication request message" $\text{Msg}_1 = \{TID_{SM_i}, X_1, R_1, X_2, \text{Cert}_{SM_i}, TS_1\}$ to SP_j via open channel.

AC2: SP_j being the responder node, after receiving the message Msg_1 at time TS_1', first checks the validity of received TS_1 by $|TS_1 - TS_1'| < \Delta T$, and if it is valid, SP_j fetches RID_{SM_i} corresponding to received TID_{SM_i} and computes $X_2' = h(TID_{SM_i} \| RID_{SM_i} \| RID_{RA} \| R_1 \| \text{Cert}_{SM_i} \| TS_1)$. If this criteria satisfies, SP_j further verifies the received certificate Cert_{SM_i} by the condition: $\text{Cert}_{SM_i}.G = \text{Pub}_{SM_i} + h(RID_{RA} \| \text{Pub}_{RA} \| RID_{SM_i}).\text{Pub}_{RA}$. If it is valid, SP_j considers SM_i as valid and proceeds to the next step.

AC3: SP_j calculates $h(TC_{SM_i} \| TS_1) = X_1 \oplus h(RID_{RA} \| RID_{SM_i} \| TS_1)$, and generates current timestamp TS_2 and random secret $r_2 \in Z_q^*$. After this, SP_j calculates $R_2 = h(r_2 \| RID_{SP_j}).G$, $DK_{ji} = h(r_2 \| RID_{SP_j}).R_1$, $Y_1 = h(TC_{SP_j} \| TS_2) \oplus h(RID_{RA} \| RID_{SM_i} \| TS_1 \| TS_2)$, the session key SK_{ji} shared with SM_i as $SK_{ji} = h(DK_{ji} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| \text{Cert}_{SM_i} \| \text{Cert}_{SP_j})$. Next, SP_j generates a new temporary identity $TID_{SM_i}^{\text{new}}$ for SM_i, and calculates the session key verifier $Y_2 = h(SK_{ji} \| TID_{SM_i}^{\text{new}} \| RID_{SM_i} \| R_2 \| \text{Cert}_{SP_j} \| TS_2)$ and $TID_{SM_i}^* = TID_{SM_i}^{\text{new}} \oplus h(TID_{SM_i} \| SK_{ji} \| RID_{SM_i} \| TS_2)$, and then sends the "node authentication response message" $\text{Msg}_2 = \{TID_{SM_i}^*, R_2, Y_1, Y_2, \text{Cert}_{SP_j}, TS_2\}$ to SM_i via open channel.

AC4: Let SM_i receive the message Msg_2 at time TS_2'. SM_i validates the received timestamp TS_2 by $|TS_2 - TS_2'| < \Delta T$. On satisfactory validation, SM_i verifies the received certificate Cert_{SP_j} by $\text{Cert}_{SP_j}.G = \text{Pub}_{SP_j} + h(RID_{RA} \| \text{Pub}_{SP_j}).\text{Pub}_{RA}$, and if it is valid, SM_i calculates $h(TC_{SP_j} \| TS_2) = Y_1 \oplus h(RID_{RA} \| RID_{SM_i} \| TS_1 \| TS_2)$, $DK_{ij} = h(r_1 \| RID_{SM_i}).R_2$, the session key SK_{ij} shared with SP_j as $SK_{ij} = h(DK_{ij} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| \text{Cert}_{SM_i} \| \text{Cert}_{SP_j})$, $TID_{SM_i}^{\text{new}} = TID_{SM_i}^* \oplus h(TID_{SM_i} \| SK_{ij} \| RID_{SM_i} \| TS_2)$ and session key verifier $Y_2' = h(SK_{ij} \| TID_{SM_i}^{\text{new}} \| RID_{SM_i} \| R_2 \| \text{Cert}_{SP_j} \| TS_2)$. If the verification $Y_2' = Y_2$ holds, SM_i generates current timestamp TS_3, calculates the session key verifier for SP_j as $X_3 = h(SK_{ij} \| TS_2 \| TS_3)$ and sends the "key establishment acknowledgement message" $\text{Msg}_3 = \{X_3, TS_3\}$ to SP_j via public medium.

AC5: If the message Msg_3 is received at time TS_3', SP_j verifies the validity of TS_3 by the condition: $|TS_3 - TS_3'| < \Delta T$. Upon satisfactory verification, SP_j calculates $X_3' = h(SK_{ji} \| TS_2 \| TS_3)$ using its previous computed SK_{ji} and generated TS_2. Now, if $X_3' = X_3$, SP_j assures that SP_j is sharing the same session key with SM_i and updates TID_{SM_i} with new $TID_{SM_i}^{\text{new}}$ in its database corresponding to SM_i. In addition, SP_j also generates a current timestamp TS_4 and computes $X_4 = h(SK_{ji} \| R_2 \| TS_4)$. SP_j then constructs a message $\text{Msg}_4 = \{X_4, TS_4\}$ and sends it to SM_i via open channel.

AC6: Assume that SM_i receives the message Msg_4 at time TS_4'. SM_i validates timestamp TS_4 by $|TS_4 - TS_4'| < \Delta T$. If it is validated successfully, SM_i checks if $X_4 = h(SK_{ij} \| R_2 \| TS_4)$. If it is valid, SM_i assures that TID_{SM_i} has been successfully updated with $TID_{SM_i}^{\text{new}}$ at SP_j, and then replaces TID_{SM_i} with $TID_{SM_i}^{\text{new}}$ in its database too. In this way, SM_i shares the same session key SK_{ij} $(= SK_{ji})$ with SP_j.

It is worth noticing that if the message Msg_3 is lost somehow during the communication, because of sending additional message Msg_4 by SP_j the smart meter SM_i will not update TID_{SM_i} with $TID_{SM_i}^{\text{new}}$ in its database. Thus, this will solve the de-synchronization in updating temporary identity in each session between SM_i and SP_j. The overall access control phase is briefed in Fig. 3.

D. Key Management Phase

Assume two service providers SP_j and SP_l agree on establishing a symmetric pairwise key between them. To achieve this purpose, the following steps are required to complete.

KM1: SP_j first generates current timestamp TS_{SP_j} and sends the request message $\{TID_{SP_j}, \text{Cert}_{SP_j}, TS_{SP_j}\}$ to SP_l via open channel.

KM2: SP_l after receiving the request message at time $TS_{SP_j}^*$, it validates timestamp by the condition: $|TS_{SP_j}^* - TS_{SP_j}| < \Delta T$. If it is verified successfully, SP_l validates the certificate by $\text{Cert}_{SP_j}.G = \text{Pub}_{SP_j} + h(RID_{RA} \| \text{Pub}_{SP_j}).\text{Pub}_{RA}$. If both checks are valid, SP_l considers SP_j as authentic and then retrieves RID_{SP_j} corresponding to TID_{SP_j} from its database and generates current timestamp TS_{SP_l}, calculates one-time pairwise shared key with SP_j as $K_{SP_j,SP_l} = h(f(RID_{SP_l}, RID_{SP_j})$

Smart meter (SM_i)	Service provider (SP_j)
Generate random secret $r_1 \in Z_q^*$, current timestamp TS_1. Compute $R_1 = h(r_1 \|\|RID_{SM_i}).G$, $X_1 = h(TC_{SM_i} \|\|TS_1) \oplus h(RID_{RA} \|\|RID_{SM_i} \|\|TS_1)$, $X_2 = h(TID_{SM_i} \|\|RID_{SM_i} \|\|RID_{RA} \|\|R_1 \|\|Cert_{SM_i} \|\|TS_1)$. $Msg_1 = \{TID_{SM_i}, X_1, R_1, X_2, Cert_{SM_i}, TS_1\}$ (via open channel)	Check if $\|TS_1 - TS_1'\| < \Delta T$? If so, fetch RID_{SM_i} corresponding to TID_{SM_i}. Compute $X_2' = h(TID_{SM_i} \|\|RID_{SM_i} \|\|RID_{RA} \|\|R_1 \|\|Cert_{SM_i} \|\|TS_1)$. Verify certificate: if $Cert_{SM_i}.G = Pub_{SM_i} + h(RID_{RA} \|\|Pub_{RA} \|\|RID_{SM_i}).Pub_{RA}$? If so, compute $h(TC_{SM_i} \|\|TS_1) = X_1 \oplus h(RID_{RA} \|\|RID_{SM_i} \|\|TS_1)$. Generate random secret $r_2 \in Z_q^*$, current timestamp TS_2. Compute $R_2 = h(r_2 \|\| RID_{SP_j}).G$, $DK_{ji} = h(r_2 \|\| RID_{SP_j}).R_1$, $Y_1 = h(TC_{SP_j} \|TS_2) \oplus h(RID_{RA} \|\|RID_{SM_i} \|\|TS_1 \|\|TS_2)$, $SK_{ji} = h(DK_{ji} \|h(TC_{SM_i} \|\|TS_1) \|\|h(TC_{SP_j} \|\|TS_2) \|\|Cert_{SM_i} \|\|Cert_{SP_j})$. Generate new temporary identity $TID_{SM_i}^{new}$. Compute $Y_2 = h(SK_{ji} \|\|TID_{SM_i}^{new} \|\|RID_{SM_i} \|\|R_2 \|\|Cert_{SP_j} \|\|TS_2)$, $TID_{SM_i}^* = TID_{SM_i}^{new} \oplus h(TID_{SM_i} \|\|SK_{ji} \|\|RID_{SM_i} \|\|TS_2)$. $Msg_2 = \{TID_{SM_i}^*, R_2, Y_1, Y_2, Cert_{SP_j}, TS_2\}$ (via open channel)
Check if $\|TS_2 - TS_2'\| < \Delta T$? Verify certificate: if $Cert_{SP_j}.G = Pub_{SP_j} + h(RID_{RA} \|\| Pub_{SP_j}).Pub_{RA}$? If so, compute $h(TC_{SP_j} \|TS_2) = Y_1 \oplus h(RID_{RA} \|\|RID_{SM_i} \|\|TS_1 \|\|TS_2)$, $DK_{ij} = h(r_1 \|\| RID_{SM_i}).R_2$, $SK_{ij} = h(DK_{ij} \|h(TC_{SM_i} \|\|TS_1) \|\|h(TC_{SP_j} \|\|TS_2) \|\|Cert_{SM_i} \|\|Cert_{SP_j})$, $TID_{SM_i}^{new} = TID_{SM_i}^* \oplus h(TID_{SM_i} \|\|SK_{ij} \|\|RID_{SM_i} \|\|TS_2)$, $Y_2' = h(SK_{ij} \|\|TID_{SM_i}^{new} \|\|RID_{SM_i} \|\|R_2 \|\|Cert_{SP_j} \|\|TS_2)$. If $Y_2' = Y_2$, generate current timestamp TS_3 and compute $X_3 = h(SK_{ij} \|\|TS_2 \|\|TS_3)$. $Msg_3 = \{X_3, TS_3\}$ (via open channel)	Check if $\|TS_3 - TS_3'\| < \Delta T$? If so, compute $X_3' = h(SK_{ji} \|\|TS_2 \|\|TS_3)$. Check if $X_3' = X_3$? If valid, generate a current timestamp TS_4. Compute $X_4 = h(SK_{ji} \|R_2 \|\|TS_4)$. $Msg_4 = \{X_4, TS_4\}$
Check if TS_4 by $\|TS_4 - TS_4'\| < \Delta T$? If so, check if $X_4 = h(SK_{ij} \|R_2 \|\|TS_4)$? If valid, update TID_{SM_i} with new $TID_{SM_i}^{new}$.	Update TID_{SM_i} with new $TID_{SM_i}^{new}$.
Both SM_i and SP_j store the shared common session key SK_{ij} $(= SK_{ji})$.	

Fig. 3. Summary of access control phase in DBACP-IoTSG.

$\|\text{Cert}_{SP_j}$ $\|\text{Cert}_{SP_l}$ $\|TS_{SP_j}$ $\|TS_{SP_l})$ and its verifier $VK_{SP_j,SP_l} = h(K_{SP_j,SP_l} \|TS_{SP_l})$. Next, SP_l sends the response message $\{TID_{SP_l}, \text{Cert}_{SP_l}, TS_{SP_l}, VK_{SP_j,SP_l}\}$ to SP_j via open channel.

KM3: Upon reception of the response message at time $TS_{SP_l}^*$, SP_j retrieves RID_{SP_l} corresponding to received TID_{SP_l} from its database, and validates timestamp by TS_{SP_l} by $|TS_{SP_l}^* - TS_{SP_l}| < \Delta T$ and the certificate Cert_{SP_l} by checking the verification equation: $\text{Cert}_{SP_l}.G = \text{Pub}_{SP_l} + h(RID_{RA} \|\text{Pub}_{SP_l}).\text{Pub}_{RA}$. If both checks are valid, SP_j calculates the one-time pairwise shared secret shared with SP_l as $K_{SP_l,SP_j} = h(f(RID_{SP_j}, RID_{SP_l}) \|\text{Cert}_{SP_j} \|\text{Cert}_{SP_l} \|TS_{SP_j} \|TS_{SP_l})$ and its verifier $VK_{SP_l,SP_j} = h(K_{SP_l,SP_j} \|TS_{SP_l})$. Note that $K_{SP_l,SP_j} = K_{SP_j,SP_l}$ as $f(RID_{SP_l}, RID_{SP_j}) = f(RID_{SP_j}, RID_{SP_l})$. If $VK_{SP_l,SP_j} = VK_{SP_j,SP_l}$, SP_j treats SP_l as authentic.

After this phase termination, both SP_j and SP_l share the same pairwise secret key K_{SP_l,SP_j} $(= K_{SP_j,SP_l})$ and use it for their secret communications in future.

E. Block Formation and Addition Phase

During the access control phase discussed in Section IV-D, it is worth noticing that an SP, say SP_j and its associated smart meter SM_i establish a session key SK_{ij} $(= SK_{ji})$. Now, using this session key SK_{ij}, SP_j will collect the encrypted informations of the form $(Tx, \text{Sign}_{Tx}, \text{BPub}_{SM_i})$ from its SMs (SM_i), where Sign_{Tx} is the ECDSA signature generation algorithm [60] and BPub_{SM_i} is public key of SM_i. Thus, SP_j can decrypt the encrypted information using the same

Block Header	
Block Version (BV_m)	Unique block version number
Previous Block Hash ($PBHash_m$)	Hash value of previous block $Block_{m-1}$
Merkle Tree Root (MTR_m)	Merkle tree root on encrypted transactions
Timestamp (TS_m)	Block creation time
Owner (O_m) of $Block_m$	A service provider (SP_j)
Public key of signer O_m	Pub_{SP_j}
Block Payload (Encrypted Transactions)	
Encrypted Transactions Tx_i	$EP_{Pub_{SP_j}}(Tx_i, Sign_{Tx_i}, BPub_{SM_i})$ $(i = 1, 2, \ldots, n_t)$
Current Block Hash ($CBHash_m$)	Hash value of current block $Block_m$ acts as a signature on block
ECDSA signature on ($CBHash_m$)	$Sign_{CBHash_m}$

Fig. 4. Formation of a block Block_m on encrypted transactions by SP_j in DBACP-IoTSG.

session key SK_{ij}. After that, SP_j forms an encrypted transaction by encrypting the decrypted information using its own public key Pub_{SP_j} as $EP_{\text{Pub}_{SP_j}}[Tx_1, \text{Sign}_{Tx_1}, \text{BPub}_{SM_i}]$ and puts it into a global transactions pool, say $\text{GTrans}_{\text{pool}}$, which will be available in the P2P SP network. Assume that $\text{GTrans}_{\text{pool}}$ is filled by a list of n_t encrypted transactions, say $\{EP_{\text{Pub}_{SP_j}}(Tx_1, \text{Sign}_{Tx_1}, \text{BPub}_{SM_i}), EP_{\text{Pub}_{SP_j}}(Tx_2, \text{Sign}_{Tx_2}, \text{BPub}_{SM_i}), \ldots, EP_{\text{Pub}_{SP_j}}(Tx_{n_t}, \text{Sign}_{Tx_{n_t}}, \text{BPub}_{SM_i})\}$. Now, when $\text{GTrans}_{\text{pool}}$ reaches to the transaction threshold, say Tx_{thresh} (the minimum number of transactions (n_t) to be stored in a block Block_m), that is, $Tx_{\text{thresh}} = n_t$, a leader (L) will be elected by Algorithm 1 from the P2P SPN using the similar strategy mentioned in [9]. After that L will create a block Block_m as mentioned in Fig. 4.

Once the block Block_m is formed by the leader L, a voting-based consensus using PBFT algorithm [28] will be executed in DBACP-IoTSG for block addition in the blockchain. First of all, the leader L sends the block Block_m along with a distinct random number to other service providers SP_j in the P2P SPN for consensus purpose of verifying the block. If a service provider SP_j verifies the block successfully with the existing $\text{GTrans}_{\text{pool}}$, it sends its verification status (*VerStatus*) securely to the leader L. L maintains the global commitment message pool (GCM_{pool}) which contains the valid block verification status (*VerStatus*) and it is accessible to all the peer nodes. Now, based on valid *VerStatus*, the leader L increments its counter (*VBCount*), where *VBCount* is the number of valid votes in the pool GCM_{pool}, which was initially set to 0. If $\text{VBCount} > 2n_f + 1$, the leader L sends *Commit* message to all the responded SPs and also adds the block Block_m in the blockchain. Meanwhile, the other peer nodes in the P2P SP network will add the block in their distributed ledger. The overall process is explained in Algorithm 2. Note that the added block Block_m can be verified by any SP and also by other entities involved in the network. However, only the service provider SP_j (L) who created Block_m can only decrypt the encrypted transactions containing in that block, because SP_j has the matching private key k_{SP_j} corresponding to the public key $\text{Pub}_{SP_j} = k_{SP_j}.G$ with the help of public key-based ECC decryption algorithm.

Remark 1: There is a pool of transactions maintained by the P2P SP network, and if it reaches to the transaction threshold, a leader (L) will be selected by the secure leader selection algorithm described in Algorithm 1. After that, the leader L will

Algorithm 1 Secure Leader Selection

1: Suppose n_f denotes the number of faulty nodes in the SP network and $n_{sp} > 3n_f + 1$.
2: Set *Follower* $\leftarrow SP_j$, $(j = 1, 2, \ldots, n_{sp})$.
3: Assume N_{ov} is the number of original votes. Set $N_{ov} \leftarrow 0$.
4: Set a random timeout RT_{out} and start a timer *TMR*.
5: **while** $TMR > RT_{out}$ **do**
6: Set *Candidate* $\leftarrow$ *Follower*.
7: Start a new timer TMR_{new}.
8: Set $N_{ov} = N_{ov} + 1$.
9: Generate a random number $r_{SP_j} \in Z_q^*$ and current timestamp TS_{SP_j}.
10: Encrypt the voting request (*VoteReq*), random number r_{SP_j} and timestamp TS_{SP_j} using the shared pairwise keys K_{SP_j,SP_l} with other service providers SP_l, $(l \neq j, j = 1, 2, \ldots, n_{sp})$ (already established during key management phase in Section IV-D) with the help of symmetric encryption $EC(\cdot)$ to generate the encrypted request $EC_{K_{SP_j,SP_l}}[VoteReq, r_{SP_j}, TS_{SP_j}]$.
11: Transmit the message $\{EC_{K_{SP_j,SP_l}}[VoteReq, r_{SP_j}, TS_{SP_j}], TS_{SP_j}\}$ to all other service providers SP_l and wait for the authentic votes reply messages.
12: Suppose SP_l receives message $\{EC_{K_{SP_j,SP_l}}[VoteReq, r_{SP_j}, TS_{SP_j}], TS_{SP_j}\}$ at time $TS^*_{SP_j}$ and wants to send vote reply message back to SP_j. SP_l first checks the validity of timestamp by the condition $|TS_{SP_j} - TS^*_{SP_j}| < \Delta T$. If this condition is verified, SP_l decrypts the encrypted request using the same pairwise key K_{SP_j,SP_l} as $(VoteReq, r'_{SP_j}, TS'_{SP_j}) = DC_{K_{SP_j,SP_l}}[EC_{K_{SP_j,SP_l}}[VoteReq, r_{SP_j}, TS_{SP_j}]]$. If the decrypted timestamp TS'_{SP_j} matches with received TS_{SP_j}, SP_l sends the vote reply message $\{EC_{K_{SP_j,SP_l}}[VoteRep, r_{SP_j}, TS_{SP_l}], TS_{SP_l}\}$ to SP_j, where TS_{SP_l} is the current timestamp generated by TS_l and *VoteRep* is the vote reply.
13: **for** each vote reply message $\{EC_{K_{SP_j,SP_l}}[VoteRep, r_{SP_j}, TS_{SP_l}], TS_{SP_l}\}$ received by SP_j from other service providers SP_l at time $TS^*_{SP_l}$ **do**
14: **if** $|TS_{SP_l} - TS^*_{SP_l}| < \Delta T$ **then**
15: Compute $(VoteRep, r^*_{SP_j}, TS^{**}_{SP_l}) = DC_{K_{SP_j,SP_l}}[EC_{K_{SP_j,SP_l}}[VoteRep, r_{SP_j}, TS_{SP_l}]]$.
16: **if** $((r^*_{SP_j} = r_{SP_j})$ and $(TS^{**}_{SP_l} = TS_{SP_l})$ and *VoteRep* is positive) **then**
17: Set $N_{ov} = N_{ov} + 1$.
18: **end if**
19: **end if**
20: **end for**
21: **if** $N_{ov} > \frac{n_{sp}}{2} + 1$ **then**
22: *Leader* $\leftarrow$ *Candidate*.
23: **else**
24: *Follower* $\leftarrow$ *Candidate*.
25: Repeat Steps 7–11 for new election.
26: **end if**
27: **end while**

Algorithm 2 Voting-Based Consensus for Block Verification and Addition in Blockchain

Input: A block $Block_m$
Output: Global commitment message pool GCM_{pool} and block addition status

1: Assume the leader (L) has the block $Block_m$. L generates a random number rn_L and current timestamp TS_L for each follower service provider, say SP_j.
2: L computes the encrypted voting request (*VotReq*) using the key K_{L,SP_j} as $EC_{K_{L,SP_j}}(VotReq, rn_L)$ and $MAC_{K_{L,SP_j}}(VotReq, rn_L, TS_L)$ using the shared key K_{L,SP_j} established during the key management phase in Section IV-D.
3: L sends $Block_m$ and MAC as $\{Block_m, EC_{K_{L,SP_j}}(VotReq, rn_L), MAC_{K_{L,SP_j}}(VotReq, rn_L, TS_L), TS_L\}$ to each follower SP_j, $(j = 1, 2, \cdots, n_{sp}, L \neq SP_j)$.
4: Assume SP_j receives the message at time TS^*_L.
5: **for** each follower node, SP_j **do**
6: **if** $(|TS_L - TS^*_L| < \Delta T)$ **then**
7: Compute block hash $CBHash'_m$ on received $Block_m$.
8: **if** $((CBHash'_m = CBHash_m)$ and $(Sign_{CBHash_m} = valid))$ **then**
9: Compute the Merkle tree root, MTR'_m on the encrypted transactions present in the block $Block_m$.
10: **if** $(MTR'_m = MTR_m)$ **then**
11: Decrypt *VotReq* using the key K_{L,SP_j} as $(VotReq', rn'_L) = DC_{K_{L,SP_j}}[EC_{K_{L,SP_j}}(VotReq, rn_L)]$ and MAC as $MAC_{K_{L,SP_j}}(VotReq, rn_L, TS_L)$.
12: **if** $(MAC_{K_{L,SP_j}}(VotReq', rn'_L, TS_L) = MAC_{K_{L,SP_j}}(VotReq, rn_L, TS_L))$ **then**
13: Send the block verification status *VerStatus* as $\{EC_{K_{L,SP_j}}(rn'_L, VerStatus)\}$ to L.
14: **end if**
15: **end if**
16: **end if**
17: **end if**
18: **end for**
19: Initialize $VBCount \leftarrow 0$.
20: **for** each received message $\{EC_{K_{L,SP_j}}(rn'_L, VerStatus)\}$ from the follower SP_j **do**
21: Compute $(rn^*_L, VerStatus) = DC_{K_{L,SP_j}}[EC_{K_{L,SP_j}}(rn'_L, VerStatus)]$.
22: **if** $((rn^*_L = rn_L)$ and $(VerStatus = valid))$ **then**
23: Set $VBCount = VBCount + 1$.
24: **end if**
25: **end for**
26: **if** $(VBCount > 2n_f + 1)$ **then**
27: Send the commit response to all followers.
28: Add block $Block_m$ to the blockchain.
29: **end if**

create a block and start the consensus algorithm in order to add this block into the blockchain as mentioned in Algorithm 2. To achieve this goal, the leader L will send the generated block with other encrypted random number and voting request to its all peer nodes. After getting the block, other follower SP nodes will verify it with the existing transactions pool. Now, assume that the leader L behaves like a malicious node, generates a fake block, and then broadcasts it to the P2P SP network. After receiving the fake block, the followers will verify the block with the existing transactions pool. If the block is found to be a fake one containing the unauthorized transactions, at the verification time other follower nodes can easily verify the received block with the existing transactions pool. However, it will not be verified with the existing pool as the block contains fake transactions. Therefore, any fake block cannot be added to the blockchain. As a result, the block transparency is achieved in the proposed scheme too.

F. Dynamic Node Addition Phase

Sometimes, some service providers SP_j may become faulty nodes or some SMs may be physically compromised by an

adversary. Thus, it becomes essential to add some new SPs or SMs in the existing IoT-enabled smart grid system.

Assume that a new smart meter SM_i^{new} needs to be installed under an existing service provider SP_j and a new service provider SP_j^{new} needs to be deployed in the existing P2P SP network. To achieve this goal, the *RA* picks for SM_i^{new} a unique real identity $\text{ID}_{SM_i}^{\text{new}}$ and a random temporary identity $TID_{SM_i}^{\text{new}}$, and computes its pseudorandom identity $RID_{SM_i}^{\text{new}} = h(\text{ID}_{SM_i}^{\text{new}} \| mk_{RA} \| RTS_{SM_i}^{\text{new}})$, where the registration timestamp of SM_i^{new} is $RTS_{SM_i}^{\text{new}}$. Furthermore, the *RA* picks a random private key $k_{SM_i}^{\text{new}}$ of SM_i^{new} and calculates the respective public key $\text{Pub}_{SM_i}^{\text{new}} = k_{SM_i}^{\text{new}}.G$, SM_i^{new}'s temporal secret $TC_{SM_i}^{\text{new}} = h(\text{ID}_{SM_i}^{\text{new}} \| mk_{RA} \| k_{SM_i}^{\text{new}} \| RTS_{SM_i}^{\text{new}})$ and certificate $\text{Cert}_{SM_i}^{\text{new}} = k_{SM_i}^{\text{new}} + h(RID_{RA} \| \text{Pub}_{RA} \| RID_{SM_i}^{\text{new}}) * mk_{RA} \pmod q$ using its own private key mk_{RA}. The *RA* picks a random private key $br_{SM_i}^{\text{new}}$ and calculates its respective public key $\text{BPub}_{SM_i}^{\text{new}} = br_{SM_i}^{\text{new}}.G$ for SM_i^{new}. The *RA* then loads the credentials $\{TID_{SM_i}^{\text{new}}, RID_{SM_i}^{\text{new}}, TC_{SM_i}^{\text{new}}, RID_{RA}, \text{Cert}_{SM_i}^{\text{new}}, (br_{SM_i}^{\text{new}}, \text{BPub}_{SM_i}^{\text{new}})\}$ in SM_i^{new}'s memory prior to its placement in the network, and declares the public keys $\text{Pub}_{SM_i}^{\text{new}}$ and $\text{BPub}_{SM_i}^{\text{new}}$ as public. In addition, the *RA* also stores the information $(TID_{SM_i}^{\text{new}}, RID_{SM_i}^{\text{new}})$ in the database of SP_j. In a similar way, the new service provider SP_j^{new} will be registered by the *RA* as described in Section IV-B2 prior to its deployment.

V. Security Analysis

This section examines the proposed DBACP-IoTSG scheme against possible attacks that are possible in blockchain-based IoT-enabled smart grid system. For this reason, we first prove the correction of established session key between a smart meter SM_i and a service provider SP_j, and then conduct the formal security analysis under the random oracle model, the informal security analysis, and also the formal security verification to assure that DBACP-IoTSG will be secure against attacks with high probability.

A. Correctness Proof of Session Key

Theorem 1 proves that the same session key is established between SM_i and SP_j.

Theorem 1: The session keys established between SM_i and SP_j are the same.

Proof: During the access control phase described in Section IV-C, the smart meter SM_i calculates the session key SK_{ij} shared with the service provider SP_j as $SK_{ij} = h(DK_{ij} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| \text{Cert}_{SM_i} \| \text{Cert}_{SP_j})$, where $DK_{ij} = h(r_1 \| RID_{SM_i}).R_2$ and $R_2 = h(r_2 \| RID_{SP_j}).G$. On the other side, SP_j also calculates the session key SK_{ji} shared with SM_i as $SK_{ji} = h(DK_{ji} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| \text{Cert}_{SM_i} \| \text{Cert}_{SP_j})$, where $R_1 = h(r_1 \| RID_{SM_i}).G$ and $DK_{ji} = h(r_2 \| RID_{SP_j}).R_1$.

Now, it follows that:

$$\begin{aligned} DK_{ij} &= h\big(r_1 \| RID_{SM_i}\big).R_2 \\ &= \big(h\big(r_1 \| RID_{SM_i}\big) * h\big(r_2 \| RID_{SP_j}\big)\big).G \\ &= h\big(r_2 \| RID_{SP_j}\big).\big(h\big(r_1 \| RID_{SM_i}\big).G\big) \\ &= h\big(r_2 \| RID_{SP_j}\big).R_1 = DK_{ji}. \end{aligned}$$

TABLE III
Queries and Their Purposes

Query	Purpose
$Execute(\Psi_{SM_i}^{l_1}, \Psi_{SP_j}^{l_2})$	Using this query, $\mathcal{A}$ is allowed to eavesdrop the messages exchanged between SM_i and SP_j
$CorruptSmartMeter(\Psi_{SM_i}^{l_1})$	Under this query, $\mathcal{A}$ is permitted to extract "the credentials stored in a stolen or lost SM_i"
$Reveal(\Psi^l)$	Under this query, the session key SK_{ij} $(= SK_{ji})$ shared between Ψ^l and its respective partner is leaked to $\mathcal{A}$
$Test(\Psi^l)$	Under this query, $\mathcal{A}$ is allowed to appeal Ψ^l for SK_{ij} $(= SK_{ji})$ and Ψ^l provides a "random outcome of a flipped unbiased coin, say c"

Hence, $SK_{ij} = h(DK_{ij} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| \text{Cert}_{SM_i} \| \text{Cert}_{SP_j}) = h(DK_{ji} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| \text{Cert}_{SM_i} \| \text{Cert}_{SP_j}) = SK_{ji}$. ■

B. Formal Security Under ROR Model

In this section, we employ the wide-accepted ROR oracle model [62] to show that DBACP-IoTSG is secure against an adversary $\mathcal{A}$ for deriving the session key between a smart meter (SM_i) and an SP (SP_j). For this goal, we briefly discuss the ROR model with semantic security notion, and then the session key security of DBACP-IoTSG in Theorem 2. The adversary $\mathcal{A}$ will have the access to all the queries tabulated in Table III. In addition, as discussed in [54], access to a "collision-resistant one-way cryptographic hash function $h(\cdot)$" is provided to all the 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: During the access control phase, two participants, namely, a smart meter (SM_i) and an SP (SP_j) are involved apart from the *RA* that is only involved during the registration and dynamic node addition phases. The notations $\Psi_{SM_i}^{l_1}$ and $\Psi_{SP_j}^{l_2}$ denote the l_1 and l_2 instances of SM_i and SP_j, respectively. These instances are known as the "random oracles."

Accepted State: An instance Ψ^l will enter in its "accepted state" once it goes to an accept state when the last valid protocol message is received. All the sent and received messages can be then ordered in sequence, and this constitutes the "session identification *sid* of Ψ^l for the current session."

Partnering: Two instances (Ψ^{l_1} and Ψ^{l_2}) are partners to each other once the following three criteria are valid.

1) Ψ^{l_1} and Ψ^{l_2} need to be in "accepted states."
2) Ψ^{l_1} and Ψ^{l_2} need to share the same *sid* and they need to also "mutually authenticate each other."
3) Ψ^{l_1} and Ψ^{l_2} need to be "mutual partners of each other."

Freshness: An instance $\Psi_{SM_i}^{l_1}$ or $\Psi_{SP_j}^{l_2}$ is called fresh when the established session key SK_{ij} $(= SK_{ji})$ shared between SM_i and SP_j is not leaked to $\mathcal{A}$ using the Reveal(Ψ^l) query described in Table III.

We now define "semantic security" of our proposed DBACP-IoTSG in Definition 1 prior to prove Theorem 2.

Definition 1 (Semantic Security): If we denote $\text{Adv}_{\mathcal{A}}^{\text{DBACP-IoTSG}}(t_{\text{poly}})$ as the "advantage of an adversary $\mathcal{A}$ running in polynomial time t_{poly} in breaking the

semantic security of the proposed DBACP-IoTSG for computing the session key SK_{ij} $(= SK_{ji})$ between a smart meter SM_i and a service provider SP_j," $\text{Adv}_{\mathcal{A}}^{\text{DBACP-IoTSG}}(t_{\text{poly}}) = |2Pr[c' = c] - 1|$, where c and c' are, respectively, the "correct" and "guessed" bits.

Theorem 2: Let there be an adversary $\mathcal{A}$ running in polynomial time t_{poly} to calculate the session key SK_{ij} $(= SK_{ji})$ established between a smart meter SM_i and a service provider SP_j in the proposed protocol, DBACP-IoTSG. If q_h, $|Hash|$, and $\text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t_{\text{poly}})$ 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 Diffe–Hellman problem (ECDDHP)," respectively, then $\text{Adv}_{\mathcal{A}}^{\text{DBACP-IoTSG}}(t_{\text{poly}}) \leq (q_h^2/|\text{Hash}|) + 2\text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t_{\text{poly}})$.

Proof: We follow the proof this theorem in a similar way that was done in [1], [54]–[56], and [63]–[66]. There are three games, say $\text{Game}_j^{\mathcal{A}}$ for the adversary $\mathcal{A}$, $j = 0, 1, 2$, where we define $\text{Succ}_{\text{Game}_j}^{\mathcal{A}}$ as "an event that $\mathcal{A}$ can guess the random bit c in the game $\text{Game}_j^{\mathcal{A}}$ correctly." We then define $\mathcal{A}$'s advantage in winning the $\text{Game}_j^{\mathcal{A}}$ in DBACP-IoTSG as $\text{Adv}_{\mathcal{A},\text{Game}_j}^{\text{DBACP-IoTSG}} = Pr[\text{Succ}_{\text{Game}_j}^{\mathcal{A}}]$. The detailed description of each game is given as follows.

$\text{Game}_0^{\mathcal{A}}$: The "actual attack" performed by the adversary $\mathcal{A}$ against the proposed DBACP-IoTSG under the ROR model always corresponds the initial game $\mathbf{Game_0^{\mathcal{A}}}$. The bit c needs to be selected randomly by $\mathcal{A}$ before the game $\text{Game}_0^{\mathcal{A}}$ begins. The semantic security defined in Definition 1 gives the following:

$$\text{Adv}_{\mathcal{A}}^{\text{DBACP-IoTSG}}(t_{\text{poly}}) = \left|2\text{Adv}_{\mathcal{A},\text{Game}_0}^{\text{DBACP-IoTSG}} - 1\right|. \quad (1)$$

$\text{Game}_1^{\mathcal{A}}$: This game corresponds to an *eavesdropping game*, where the adversary $\mathcal{A}$ makes use of the defined *Execute* query in Table III. Using this query, $\mathcal{A}$ will be able to intercept all the communicated messages $\text{Msg}_1 = \{TID_{SM_i}, X_1, R_1, X_2, \text{Cert}_{SM_i}, TS_1\}$, $\text{Msg}_2 = \{TID^*_{SM_i}, R_2, Y_1, Y_2, \text{Cert}_{SP_j}, TS_2\}$, $\text{Msg}_3 = \{X_3, TS_3\}$ and $\text{Msg}_4 = \{X_4, TS_4\}$, and try to derive the session key SK_{ij} $(= SK_{ji})$. Next, $\mathcal{A}$ needs to execute *Reveal* and *Test* queries in order to check whether the derived session key is a correct one or just a random key. It is worth noting that $SK_{ij} = h(DK_{ij}\|h(TC_{SM_i}\|TS_1)\|h(TC_{SP_j}\|TS_2)\|\text{Cert}_{SM_i}\|\text{Cert}_{SP_j}) = h(DK_{ji}\|h(TC_{SM_i}\|TS_1)\|h(TC_{SP_j}\|TS_2)\|\text{Cert}_{SM_i}\|\text{Cert}_{SP_j}) = SK_{ji}$, where $DK_{ij} = h(r_1\|RID_{SM_i}).R_2 = (h(r_1\|RID_{SM_i}) * h(r_2\|RID_{SP_j})).G = h(r_2\|RID_{SP_j}).(h(r_1\|RID_{SM_i}).G) = h(r_2\|RID_{SP_j}).R_1 = DK_{ji}$. Since all the temporal and long term secrets are protected by $h(\cdot)$, only interception of the messages Msg_m $(m = 1, 2, 3, 4)$ will not lead to increase the success probability at all in deriving the session key SK_{ij} $(= SK_{ji})$. Now, both the games $\text{Game}_0^{\mathcal{A}}$ and $\text{Game}_1^{\mathcal{A}}$ become indistinguishable under the eavesdropping attack. Thus, we obtain the following:

$$\text{Adv}_{\mathcal{A},\text{Game}_1}^{\text{DBACP-IoTSG}} = \text{Adv}_{\mathcal{A},\text{Game}_0}^{\text{DBACP-IoTSG}}. \quad (2)$$

$\text{Game}_2^{\mathcal{A}}$: This game will correspond to an active attack, where we include the simulations of *Hash* and *CorruptSmartMeter* queries, and also difficulty of solving ECDDHP. To derive the session key SK_{ij} $(= SK_{ji})$, the adversary $\mathcal{A}$ needs to derive DK_{ij} $(= DK_{ji})$, where $DK_{ij} = h(r_1\|RID_{SM_i}).R_2$ and $DK_{ji} = h(r_2\|RID_{SP_j}).R_1$. Assume that $\mathcal{A}$ has already the intercepted messages Msg_m $(m = 1, 2, 3, 4)$, and so, he/she has the knowledge of $R_1 = h(r_1\|RID_{SM_i}).G$ and $R_2 = h(r_2\|RID_{SP_j}).G$. Since $DK_{ij} = h(r_1\|RID_{SM_i}).R_2 = (h(r_1\|RID_{SM_i}) * h(r_2\|RID_{SP_j})).G = h(r_2\|RID_{SP_j}).(h(r_1\|RID_{SM_i}).G) = h(r_2\|RID_{SP_j}).R_1 = DK_{ji}$, the adversary $\mathcal{A}$ has to solve the computational ECDDHP to obtain DK_{ij} $(= DK_{ji})$ using R_1 and R_2. Furthermore, other secrets (TC_{SM_i} and TC_{SP_j}) are embedded in the hash function $h(\cdot)$. In addition, using the *CorruptSmartMeter* query, the adversary $\mathcal{A}$ will have the credentials $\{TID_{SM_i}, RID_{SM_i}, TC_{SM_i}, RID_{RA}, \text{Cert}_{SM_i}\}$. Then, having the knowledge of other secrets, such as r_1, r_2, RID_{SP_j} and TC_{SP_j}, $\mathcal{A}$ will be able to derive the session key SK_{ij} $(= SK_{ji})$. We observe that both the games $\text{Game}_1^{\mathcal{A}}$ and $\text{Game}_2^{\mathcal{A}}$ are indistinguishable if we do not have simulation of *Hash* and *CorruptSmartMeter* queries, and ECDDHP is not hard problem. Using the results of birthday paradox for finding the hash collision and the advantage of solving ECDDHP, we obtain the following relation:

$$\left|\text{Adv}_{\mathcal{A},\text{Game}_1}^{\text{DBACP-IoTSG}} - \text{Adv}_{\mathcal{A},\text{Game}_2}^{\text{DBACP-IoTSG}}\right| \leq \frac{q_h^2}{2|\text{Hash}|} + \text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t_{\text{poly}}). \quad (3)$$

It is worth noting that all the queries are made by $\mathcal{A}$, and it is only left for $\mathcal{A}$ to correctly guess a bit to win the game $\text{Game}_2^{\mathcal{A}}$. Therefore, we have

$$\text{Adv}_{\mathcal{A},\text{Game}_2}^{\text{DBACP-IoTSG}} = \frac{1}{2}. \quad (4)$$

Equation (1) gives

$$\frac{1}{2}.\text{Adv}_{\mathcal{A}}^{\text{DBACP-IoTSG}}(t_{\text{poly}}) = \left|\text{Adv}_{\mathcal{A},\text{Game}_0}^{\text{DBACP-IoTSG}} - \frac{1}{2}\right|. \quad (5)$$

Equations (2)–(4), and use of triangular inequality lead to the following derivation from (5):

$$\begin{aligned}\frac{1}{2}.&\text{Adv}_{\mathcal{A}}^{\text{DBACP-IoTSG}}(t_{\text{poly}}) \\ &= \left|\text{Adv}_{\mathcal{A},\text{Game}_0}^{\text{DBACP-IoTSG}} - \text{Adv}_{\mathcal{A},\text{Game}_2}^{\text{DBACP-IoTSG}}\right| \\ &= \left|\text{Adv}_{\mathcal{A},\text{Game}_1}^{\text{DBACP-IoTSG}} - \text{Adv}_{\mathcal{A},\text{Game}_2}^{\text{DBACP-IoTSG}}\right| \\ &\leq \frac{q_h^2}{2|\text{Hash}|} + \text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t_{\text{poly}}). \quad (6)\end{aligned}$$

Finally, if we multiply both sides of (6) by "a factor of 2," we arrive to the final result: $\text{Adv}_{\mathcal{A}}^{\text{DBACP-IoTSG}}(t_{\text{poly}}) \leq (q_h^2/|\text{Hash}|) + 2\text{Adv}_{\mathcal{A}}^{\text{ECDDHP}}(t_{\text{poly}})$. ■

C. Informal Security Analysis

Through the informal (nonmathematical) security analysis, we exhibit that the proposed DBACP-IoTSG resists various attacks, which are proved in Propositions 1–7.

Proposition 1: DBACP-IoTSG is secure against smart meter and SP impersonation attacks.

Proof: We consider the following impersonation attacks related to our DBACP-IoTSG.

Smart Meter Impersonation Attack: Suppose an adversary $\mathcal{A}$ acts as a register smart meter SM_i and wants to communicate with the service provider SP_j with the message $Msg'_1 = \{TID_{SM_i}, X'_1, R'_1, X'_2, Cert_{SM_i}, TS'_1\}$. To do so, $\mathcal{A}$ can select a random number r'_1 and timestamp TS'_1 to calculate $R'_1 = h(r'_1 \| RID_{SM_i}).G$ and $X'_1 = h(TC_{SM_i} \| TS'_1) \oplus h(RID_{RA} \| RID_{SM_i} \| TS'_1)$ and $X'_2 = h(TID_{SM_i} \| RID_{SM_i} \| RID_{RA} \| R'_1 \| Cert_{SM_i} \| TS'_1)$. Since RID_{SM_i} and TC_{SM_i} are the secret credentials, so without knowledge of these credentials it is "computationally infeasible problem" for $\mathcal{A}$ to generate $R'_1 = h(r'_1 \| RID_{SM_i}).G$, $X'_1 = h(TC_{SM_i} \| TS'_1) \oplus h(RID_{RA} \| RID_{SM_i} \| TS'_1)$ and $X_2 = h(TID_{SM_i} \| RID_{SM_i} \| RID_{RA} \| R'_1 \| Cert_{SM_i} \| TS'_1)$ on behalf of smart meter SM_i. Hence, DBACP-IoTSG is resilient against "smart meter impersonation attack."

Service Provider Impersonation Attack: Let an adversary $\mathcal{A}$ act as a valid service provider SP_j and want to send a valid message $Msg'_2 = \{(TID^*_{SM_i})', R'_2, Y'_1, Y'_2, Cert_{SP_j}, TS'_2\}$ to SM_i. For this purpose, $\mathcal{A}$ can select a random number r'_2 and timestamp TS'_2 to generate $R'_2 = h(r'_2 \| RID_{SP_j}).G$, $DK'_{ji} = h(r'_2 \| RID_{SP_j}).R_1$ and $Y'_1 = h(TC_{SP_j} \| TS'_2) \oplus h(RID_{RA} \| RID_{SM_i} \| TS'_1 \| TS'_2)$. $\mathcal{A}$ can also try to calculate the session key $SK'_{ji} = h(DK'_{ji} \| h(TC_{SM_i} \| TS'_1) \| h(TC_{SP_j} \| TS'_2) \| Cert_{SM_i} \| Cert_{SP_j})$ and generate a new temporal identity $(TID^{new}_{SM_i})'$ to calculate session key verifier $Y'_2 = h(SK'_{ji} \| (TID^{new}_{SM_i})' \| RID_{SM_i} \| R'_2 \| Cert_{SP_j} \| TS'_2)$ and $(TID^*_{SM_i})' = (TID^{new}_{SM_i})' \oplus h(TID_{SM_i} \| SK'_{ji} \| RID_{SM_i} \| TS'_2)$. Again it is "computationally infeasible problem" for $\mathcal{A}$ to generate R'_2, DK'_{ji}, Y'_1, SK'_{ji}, Y'_2, and $(TID^*_{SM_i})'$ without knowledge of the secrete credentials RID_{SP_j}), TC_{SP_j}, RID_{SM_i}, and TC_{SM_i}. Hence, DBACP-IoTSG is resilient against "SP impersonation attack." ■

Proposition 2: DBACP-IoTSG is resilient against smart meter physical capture attack.

Proof: Due to unfriendly (unattended) environment some smart meters may be captured physically by an adversary $\mathcal{A}$. Then, $\mathcal{A}$ can extract the all stored information $\{TID_{SM_i}, RID_{SM_i}, TC_{SM_i}, RID_{RA}, Cert_{SM_i}\}$ from a captured smart meter SM_i which were stored during smart meter registration phase (see Section IV-B1) using the power analysis attacks [52]. Since the secret credentials $\{TID_{SM_i}, RID_{SM_i}, TC_{SM_i}, Cert_{SM_i}\}$ are distinct for different smart meters, compromising these credentials can not effect to the whole network. This is primarily because these credentials will not be helpful in computing the session keys between other noncompromised smart meters and a service provider SP_j. Thus, compromise of SM_i does not reflect in compromising secure communications among noncompromised smart meters SM'_i and a service provider SP_j, and hence, they can still communicate with 100% security. This assures that DBACP-IoTSG is "unconditionally secure against smart meters capture attack" and as a result, it is protected from "smart meter physical capture attack." ■

Proposition 3: Replay attack is protected in DBACP-IoTSG.

Proof: The messages $Msg_1 = \{TID_{SM_i}, X_1, R_1, X_2, Cert_{SM_i}, TS_1\}$, $Msg_2 = \{TID^*_{SM_i}, R_2, Y_1, Y_2, Cert_{SP_j}, TS_2\}$, $Msg_3 = \{X_3, TS_3\}$ and $Msg_4 = \{X_4, TS_4\}$ are sent over public channel during the "access control phase" discussed in Section IV-D, which is happened in between smart meter SM_i and service provider SP_j. At the time of message creation, not only the timestamps are included but the random numbers are also attached along with the messages. The timestamps are verified by the receiver(s) for checking message integrity and freshness. Since the adversary $\mathcal{A}$ cannot produce the original timestamps which are used in that time attached in the messages, replaying the old valid messages are then detected by the receivers. Thus, DBACP-IoTSG is protected from the "replay attack." ■

Proposition 4: The man-in-the-middle attack is protected in DBACP-IoTSG.

Proof: In this attack, an adversary $\mathcal{A}$ may intercept the "node authentication request message" $Msg_1 = \{TID_{SM_i}, X_1, R_1, X_2, Cert_{SM_i}, TS_1\}$ from an insecure (open) channel and generate another valid message Msg'_1 on the fly so that the service provider SP_j as the receiver can not detect it as a modified one. But, $\mathcal{A}$ can not generate the values of R_1, X_1, and X_2 to produce the valid message Msg'_1 due to the preloaded secrete credentials TC_{SM_i} and RID_{SM_i}. In a similar way, $\mathcal{A}$ also fails to create valid "node authentication response message" for the intercepted message $Msg_2 = \{TID^*_{SM_i}, R_2, Y_1, Y_2, Cert_{SP_j}, TS_2\}$ due to preloaded secret information TC_{SP_j} and RID_{SP_j}, and the shared session secrete key SK_{ji} is needed for this purpose. Furthermore, $\mathcal{A}$ can not temper the "key establishment acknowledgment message" $Msg_3 = \{X_3, TS_3\}$ and Msg_4 due to computation of the authentic secret shared session key SK_{ji}. Thus, DBACP-IoTSG is secure against the "man-in-the-middle attack." ■

Proposition 5: DBACP-IoTSG is resilient against the ESL attack.

Proof: In DBACP-IoTSG, using the access control phase, the smart meter SM_i calculates the session key SK_{ij} shared with the service provider SP_j as $SK_{ij} = h(DK_{ij} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| Cert_{SM_i} \| Cert_{SP_j})$, where $DK_{ij} = h(r_1 \| RID_{SM_i}).R_2$ and $R_2 = h(r_2 \| RID_{SP_j}).G$. On the other side, SP_j also calculates the session key SK_{ji} shared with SM_i as $SK_{ji} = h(DK_{ji} \| h(TC_{SM_i} \| TS_1) \| h(TC_{SP_j} \| TS_2) \| Cert_{SM_i} \| Cert_{SP_j})$, where $R_1 = h(r_1 \| RID_{SM_i}).G$ and $DK_{ji} = h(r_2 \| RID_{SP_j}).R_1$. Since $DK_{ji} = DK_{ij}$, both SM_i and SP_j share the same session key $SK_{ij} = (= SK_{ji})$, which is also proved in Theorem 1. It is understandable that the "session key" is the combination of both the session-temporary (ephemeral) credentials (also called as "short term secrets"), such as random numbers and the "long-term secrets" (different secret credentials and pseudoidentities). The session key SK_{ij} can only be disclosed when an adversary $\mathcal{A}$ compromises both the session-temporary as well as long-term secrets. Moreover, usage of random numbers and timestamps in computation of session keys between various SM_i and SP_j over different sessions makes distinct session keys establishment among SM_i and SP_j. Even if a session key is disclosed for a specific session, it will not result in calculating the session keys over other sessions because short and long term secrets are used. As a result, DBACP-IoTSG is secure against "session-temporary information attack" and it also preserves the "perfect forward secrecy" property. In a nutshell, DBACP-IoTSG is secure against "ESL attack." ■

Proposition 6: DBACP-IoTSG preserves both anonymity and untraceability functionalities.

Proof: Suppose an adversary $\mathcal{A}$ eavesdrops the messages Msg_1, Msg_2, Msg_3 and Msg_4. Since each of the messages does not contain the real identity ID_{SM_i} of a smart meter SM_i and the real identity ID_{SP_j} of a service provider SP_j directly, $\mathcal{A}$ cannot relate who is the sender or receiver in the node authentication and key establishment session in access control phase. Therefore, "anonymity" of both smart meter and SP is preserved in DBACP-IoTSG. Again, the parameters involved in various messages Msg_1, Msg_2, Msg_3 and Msg_4 are purely dynamic, and these are not same for any two key establishment sessions in access control phase due to usage of both random numbers and current timestamps. Hence, $\mathcal{A}$ cannot relate whether the messages exchanged between the entities over two successive sessions belong to the same user or not. This also assures "untraceability" in DBACP-IoTSG. ■

Proposition 7: Block verification is supported in DBACP-IoTSG.

Proof: In DBACP-IoTSG, the block verification by the peer nodes (SPs) in the P2P SP network is based on the well-known voting-based PBFT consensus algorithm as discussed in Algorithm 2. It is worth noticing that a smart grid system is considered as an "asynchronous distribution system." Thus, a single node (peer node) failure is treated as an independent event. Suppose an adversary $\mathcal{A}$ controls some nodes in the SP network and enables them with malicious consensus algorithm. However, in DBACP-IoTSG, the leader (L) makes a decision based on responses (erroneous response or positive response or no response). As long as the L receives an adequate number of replies from the "nonfailed nodes," DBACP-IoTSG assures security and activity of asynchronous system to encounter the requirements. In addition, the leader selection in DBACP-IoTSG takes place securely through the use of established pairwise secret keys among the SPs acting as peer nodes in the SP network as illustrated in Algorithm 1. As a result, there is a negligle possibility of getting corrupted responses from the nonfailed authentic nodes. Thus, in DBACP-IoTSG, the block verification takes place in a secure manner. ■

D. Formal Security Verification: Simulation Study Using AVISPA

This section gives the formal security verification of the proposed DBACP-IoTSG scheme using the broadly used automated software verification tool, known as "AVISPA" [30]. In recent years, AVISPA-based simulation becomes reliable as it has been applied in many authentication and access control protocols, such as the schemes in [1], [53], and [55]–[57].

AVISPA is a "push button tool for formal verification," which can detect whether a security protocol is "safe," "unsafe," or "inconclusive" against passive and active attacks. Presently, AVISPA has the ability to detect replay and man-in-the-middle attacks by analyzing a security protocol through the formal verification. To implemented a tested security protocol, the protocol needs to implement using the "high-level protocol specification language (HLPSL)." The HLPSL code written in a file with extension *·hlpsl* is translated into the "intermediate format (IF)" using the "HLPSL2IF" translator. The IF is then

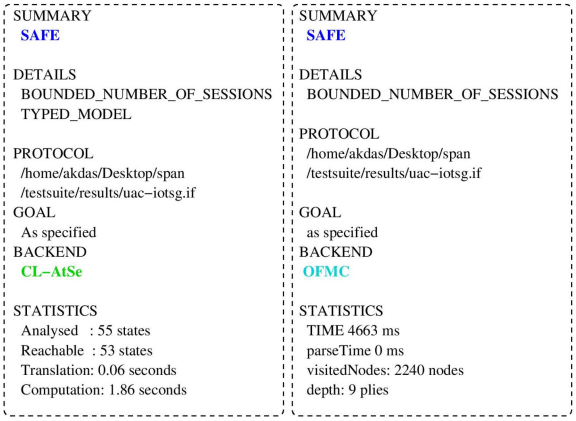

Fig. 5. Simulation results of DBACP-IoTSG under CL-AtSe and OFMC backends.

provided into one of the available four backends, namely: "on-the-fly mode-checker (OFMC)," "constraint-logic-based attack searcher (CL-AtSe)," "SAT-based model checker (SATMC)" and "tree automata based on automatic approximations for the analysis of security protocols (TA4SP)." Currently, SATMC and TA4SP backends do not support "bitwise exclusive OR (XOR)" operation. Therefore, in this simulation, we only stick on two backends: 1) OFMC and 2) CL-AtSe. The interested readers can see all the details about AVISPA and HLPSL implementation in [30].

In our implementation, we have mainly three basic roles for the *RA*, a smart meter SM_i and a service provider SP_j. Apart from these three basic roles, two mandatory roles for the session and goal & environment need to be implemented. The basic purpose of defining the role of goal is to keep a check on authentication of the communicated messages along with confidentiality of the secret variables defined in the proposed DBACP-IoTSG.

We have then simulated DBACP-IoTSG using the "security protocol animator for AVISPA (SPAN)" tool [67]. The simulation results are shown in Fig. 5. It is worth noticing that AVISPA implements the "DY threat model (DY model)" [51]. Hence, an intruder (in AVISPA, it is always denoted by *i*) not only can intercept the communicated messages, but can also modify, delete or insert the malicious messages in between the communication. Under OFMC backend, the simulation required 4663 ms with 2240 visited nodes and 9 plies of depth. Under CL-AtSe backend, the simulation analyzed 55 states and out of these states, 53 states were reachable, and it took translation time of 0.06 s and computation time of 1.86 s. The results in Fig. 5 clearly indicate that the proposed DBACP-IoTSG is secure against replay and man-in-the-middle attacks.

VI. Experimental Results Using MIRACL

In this section, we provide the experimental results of various cryptographic primitives that are needed for comparative analysis in Section VII using the widely used MIRACL [31]. MIRACL, a C programming-based software library, is widely accepted by the developers as the "gold standard open source SDK for ECC" [31].

TABLE IV
EXPERIMENTAL EXECUTION TIME (IN MILLISECONDS) FOR SP

Primitive	Max. time (ms)	Min. time (ms)	Average time (ms)
T_h	0.149	0.024	0.055
T_{exp}	0.248	0.046	0.072
T_{mtp}	0.199	0.092	0.114
T_{ecm}	2.998	0.284	0.674
T_{eca}	0.002	0.001	0.002
T_{ecenc}	5.998	0.569	1.350
T_{ecdec}	3.000	0.285	0.676
T_{bp}	7.951	4.495	4.716

TABLE V
EXECUTION TIME (IN MILLISECONDS) FOR SMART METER (RASPBERRY PI 3)

Primitive	Max. time (ms)	Min. time (ms)	Average time (ms)
T_h	0.643	0.274	0.309
T_{exp}	0.071	0.037	0.039
T_{mtp}	0.406	0.381	0.385
T_{ecenc}	9.085	4.427	4.592
T_{ecdec}	4.553	2.221	2.304
T_{ecm}	4.532	2.206	2.288
T_{eca}	0.021	0.015	0.016
T_{bp}	32.79	27.606	32.084

We perform the following cryptographic operations using MIRACL. The symbols T_{bp}, T_{ecm}, T_{eca}, T_{ecenc}/T_{ecdec}, T_{mtp}, T_{exp}, and T_h are used to denote the time required for "bilinear pairing," "elliptic curve point (scalar) multiplication," "elliptic curve point addition," "elliptic curve encryption/decryption," "map to elliptic curve point," "modular exponentiation" and "one-way hash function using SHA-256 hashing algorithm," respectively. Moreover, a nonsingular elliptic curve of the form: "$y^2 = x^3 + ax + b \pmod q$ with $4a^3 + 27b^2 \neq 0 \pmod q$" has been considered.

We consider the following two platforms.

Platform 1: In this case, the platform is considered on the setting: "Ubuntu 18.04.4 LTS, with memory: 7.7 GiB, processor: Intel Core i7-8565U CPU @ 1.80 GHz × 8, OS type: 64-bit and disk: 966.1 GB." Each experiment for a cryptographic primitive is run for 100 times. Next, we have calculated the maximum, minimum and average run-time in milliseconds required for each cryptographic primitive from these 100 runs. Table IV tabulates the experimental results for the considered cryptographic primitives.

Platform 2: In this case, we consider the platform on the setting: "Raspberry PI 3 B+ Rev 1.3, Ubuntu 20.04 LTS, 64-b OS, 1.4-GHz Quad-core processor, cores 4, 1-GB RAM [68]." Similar to the above case, we have also executed each primitive for 100 runs, and from these runs the maximum, minimum and average execution time in milliseconds are computed. The experimental results for each primitive are then reported in Table V.

VII. COMPARATIVE ANALYSIS

This section gives a detailed comparative analysis among our proposed DBACP-IoTSG and other recent schemes of Zhang *et al.* [9] and Zhou *et al.* [42]. We measure the performance comparisons of DBACP-IoTSG with other schemes [9], [42] for communication and computation costs, and also for security and functionality attributes.

A. Security and Functionality Attributes Comparison

We compare various security and functionality attributes among our proposed DBACP-IoTSG and other relevant schemes that are also based on blockchain technology [9], [42]. Table VI exhibits the comparative study on these features among the schemes. It is worth noticing that our DBACP-IoTSG provides much better security and also supports more functionality features as compared to other schemes. Zhang *et al.*'s scheme [9] does not support "anonymity" and "untraceability" features which are treated as crucial features in the IoT-enabled smart grid environment. The consensus algorithms used in the Zhou *et al.*'s scheme [42] and Zhang *et al.*'s scheme [9] are FBFT and PBFT, respectively. In our DBACP-IoTSG, we have utilized the puzzle-based solution combined with PBFT in order to make consensus more secure with the pairwise keys established among the SPs.

B. Communication Costs Comparison

For comparative study on communication costs, it is assumed that 160-bit ECC provides the same security level while it is compared with 1024-bit RSA public key cryptosystem, and thus, an "elliptic curve point of the form $G = (G_x, G_y)$, where x and y co-ordinates of G are G_x and G_y respectively," requires $(160 + 160) = 320$ b. Since the random numbers are selected from the finite field $GF(q)$, they are 160 b. Furthermore, a timestamp is 32 b and a hash value (digest) is 256 b, if SHA-256 hash algorithm [59] is utilized for the blockchain technology to provide sufficient security level, and a message size in all schemes is taken as 160 b. Table VII shows comparison of communication costs among the schemes with the number of messages and the number of bits required during the access control phase. In our DBACP-IoTSG, four exchanged messages $\text{Msg}_1 = \{TID_{SM_i}, X_1, R_1, X_2, \text{Cert}_{SM_i}, TS_1\}$, $\text{Msg}_2 = \{TID^*_{SM_i}, R_2, Y_1, Y_2, \text{Cert}_{SP_j}, TS_2\}$, $\text{Msg}_3 = \{X_3, TS_3\}$, and $\text{Msg}_4 = \{X_4, TS_4\}$ demand 1184, 1280, 288, and 288 b, a total cost of 3040 b. It is seen that our DBACP-IoTSG requires less communication costs as compared to the scheme designed by Zhang *et al.* [9]. Although the scheme of Zhou *et al.* [42] needs less communication cost as compared to our DBACP-IoTSG, it is acceptable because it does not also support all the security and functionality attributes (see Table VI) as compared to DBACP-IoTSG.

C. Computational Costs Comparison

We mainly consider the computation costs comparison during the access control phase among the proposed DBACP-IoTSG and other existing competing schemes. We use the experimental results reported in Table IV for an SP or a power provider or cloud storage provider. On the other side, since a smart meter or a user's mobile device or smart card is resource constrained as compared to a server, we consider the experimental results reported in Table V for the smart meter or user. In DBACP-IoTSG, a smart meter SM_i requires computational cost of $4T_{ecm} + T_{eca} + 11T_h$, which is roughly 12.567 ms, whereas an SP's computational cost is $4T_{ecm} + T_{eca} + 11T_h$,

TABLE VI
SECURITY AND FUNCTIONALITY ATTRIBUTES COMPARISON

Scheme	$SFA1$	$SFA2$	$SFA3$	$SFA4$	$SFA5$	$SFA6$	$SFA7$	$SFA8$	$SFA9$	$SFA10$	$SFA11$	$SFA12$	$SFA13$	$SFA14$	$SFA15$
Zhou *et al.* [42]	✓	✓	✓	✓	✓	×	×	×	×	✓	✓	✓	×	FBFT	×
Zhang *et al.* [9]	×	✓	✓	✓	✓	×	×	×	×	×	×	✓	×	PBFT	×
DBACP-IoTSG	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	Voting-based PBFT	✓

Note: $SFA1$: "impersonation attacks"; $SFA2$: "replay attack"; $SFA3$: "man-in-the-middle attack"; $SFA4$: "mutual authentication between a smart meter and service provider/power provider/cloud storage provider" without involvement of a trusted authority; $SFA5$: "smart meter physical capture attack"; $SFA6$: "session key-security"; $SFA7$: "ESL attack"; $SFA8$: "perfect forward secrecy"; $SFA9$: "dynamic smart meter/service provider addition phase"; $SFA10$: "anonymity preservation"; $SFA11$: "untraceability preservation"; $SFA12$: "support blockchain-based solution"; $SFA13$: "secure leader selection in the P2P SP network"; $SFA14$: "type of consensus mechanism used"; $SFA15$: "secure voting during consensus algorithm"
✓: "the scheme is secure or it supports a feature"; ×: "the scheme is insecure or it does not support a feature"; FBFT: "Federated Byzantine Fault Tolerance"; PBFT: "Practical Byzantine Fault Tolerance"

TABLE VII
COMMUNICATION COSTS COMPARISON

Scheme	No. of messages	No. of bits
Zhou *et al.* [42]	3	2464
Zhang *et al.* [9]	4	3328
DBACP-IoTSG	4	3040

TABLE VIII
COMPUTATION COSTS COMPARISON

Scheme	Smart meter/User side	Service provider/Power provider /Cloud storage provider side
Zhou *et al.* [42]	$3T_{ecm} + T_{eca} + T_{mtp}$ $+3T_{bp} + 2T_h$ ≈ 104.135 ms	$6T_{ecm} + T_{eca} + T_{mtp} + 5T_{bp}$ $+T_{exp} + 8T_h$ ≈ 28.252 ms
Zhang *et al.* [9]	$4T_h$ ≈ 1.236 ms	$T_{ecenc} + T_h$ ≈ 1.405 ms
DBACP-IoTSG	$4T_{ecm} + T_{eca} + 11T_h$ ≈ 12.567 ms	$4T_{ecm} + T_{eca} + 11T_h$ ≈ 3.303 ms

which is roughly 3.303 ms. It is evident from Table VIII that DBACP-IoTSG needs less computational costs as compared to Zhou *et al.*'s scheme [42]. Although Zhang *et al.*'s scheme [9] demands less computational cost as compared to both Zhou *et al.*'s scheme [42] and DBACP-IoTSG, it requires more communication cost (see Table VII) and it does not also support all the security and functionality attributes (see Table VI).

VIII. BLOCKCHAIN IMPLEMENTATION

In this section, we provide the blockchain implementation of the proposed scheme (DBACP-IoTSG). The simulations were executed on a platform having "Ubuntu 18.04, 64-b OS with Intel Core i5-4210U CPU @ 1.70 GHz, 4-GB RAM." The script was developed in the Node.js language with VS CODE 2019 [69].

We considered that the number of peer nodes (n_{sp}) in the P2P SP network is 20. A SP is a part of the P2P SP network, that securely collects the transactions from associated smart meters to form a global transactions pool in the network. If the number of transaction in transactions pool reaches to a predefined transaction threshold (the minimum number of transaction to stored in a block), a leader is selected from the network for creating a block and adding that block in the blockchain. The leader, say, L creates a block Block_m which has the structure as shown in Fig. 4, and wants to add this block into the blockchain. After executing the consensus algorithm using the "PBFT consensus algorithm" [28] provided in Algorithm 2, a block is finally added by the leader L in the blockchain.

We consider the block version, previous block hash, Merkle tree root, timestamp (epoch time), owner (O_m) of Block_m that is owner identity, public key of O_m, block payload, current block hash (CBHash_m) (using the SHA-256 hashing algorithm), and ECDSA signature on $CBHash_m$ are of sizes 32, 256, 256, 42, 160, 320, $640n_t$, 256, and 320 b, respectively. Each transaction Tx_i was encrypted using ECC encryption which outputs two elliptic curve points, and as a result, an encrypted transaction requires $(320 + 320) = 640$ b. As a result, the total size for a block Block_m becomes $1642 + 640n_t$ b.

In the following, three types of simulation cases are considered.

1) *Case 1:* In this case, we assume $n_{sp} = 20$ and the number of transactions per block is 70. The simulation results shown in Fig. 6 shows the number of blocks mined into the blockchain versus the total computational time (in seconds). It is noticed that as the number of blocks mined is increased, the computational time also increases.
2) *Case 2:* In this scenario, we assume $n_{sp} = 20$ and the number of mined blocks in each chain is 60. The simulation results are provided in Fig. 6. The simulation results are based on the number of transactions pushed per block versus the total computational time (in seconds). Similar to the trends observed in case 1, the computational time also increases as the number of transactions per block is increased.
3) *Case 3:* In this scenario, we have fixed the number blocks mined as 40 and the number of transactions per block as 50. The simulation results provided for case 3 in Fig. 6 shows the computational time increases when the number of peer nodes (n_{sp}) in the P2P network is increased.

IX. CONCLUSION

In this work, we attempted to address an important research topic in the IoT-enabled smart grid system by designing a novel DBACP-IoTSG. The proposed DBACP-IoTSG works without involving a TTP. The blocks are then verified by a leader selection process securely in the P2P SP network. After that the leader is responsible for running the consensus algorithm securely to validate the blocks by its peer

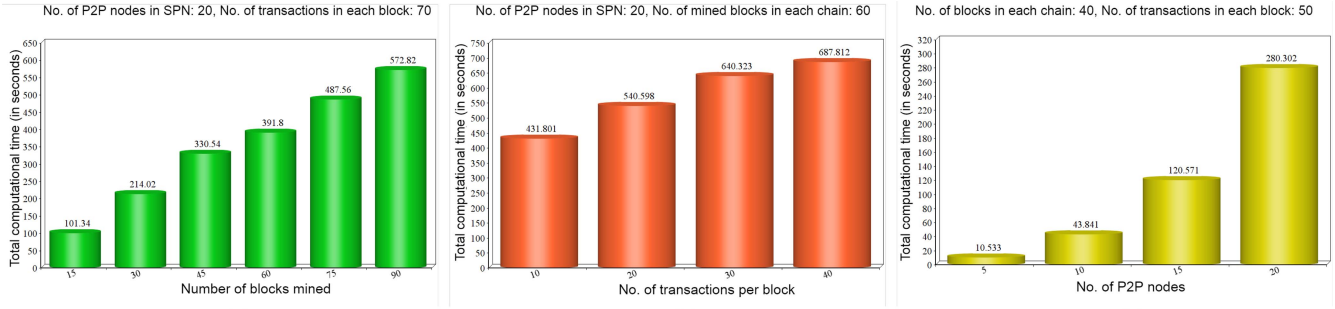

Fig. 6. Blockchain simulation results for cases 1–3.

nodes using the PBFT method, and after successful validation the blocks are added into the blockchain. The transactions are encrypted using ECC encryption algorithm using the public key Pub_{SP_j} of a service provider SP_j who generates the blocks containing the transactions so that only SP_j can decrypt the transactions. DBACP-IoTSG is shown to be secure through a rigorous security analysis using formal, informal and simulation-based formal security verification. A thorough comparative study among the proposed DBACP-IoTSG and other relevant schemes shows DBACP-IoTSG's better security and supporting of more functionality attributes. Moreover, DBACP-IoTSG is also comparable with other schemes in terms of communication and computation costs. Finally, the blockchain-based implementation for the proposed DBACP-IoTSG has been carried out to measure the computational time required for the varied number of blocks in the blockchain and also the varied number of transactions per block.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers for their valuable feedback on the article, which helped them to improve its quality and presentation.

REFERENCES

[1] M. Wazid, A. K. Das, V. Odelu, N. Kumar, and W. Susilo, "Secure remote user authenticated key establishment protocol for smart home environment," *IEEE Trans. Depend. Secure Comput.*, vol. 17, no. 2, pp. 391–406, Mar./Apr. 2020.

[2] K. Kim and P. R. Kumar, "Cyber-physical systems: A perspective at the centennial," *Proc. IEEE*, vol. 100, pp. 1287–1308, May 2012.

[3] R. Alvaro-Hermana, J. Fraile-Ardanuy, P. J. Zufiria, L. Knapen, and D. Janssens, "Peer to peer energy trading with electric vehicles," *IEEE Intell. Transp. Syst. Mag.*, vol. 8, no. 3, pp. 33–44, 2016.

[4] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Trans. Depend. Secure Comput.*, vol. 15, no. 5, pp. 840–852, Oct./Sep. 2018.

[5] S. Aggarwal, R. Chaudhary, G. S. Aujla, N. Kumar, K. K. R. Choo, and A. Y. Zomaya, "Blockchain for smart communities: Applications, challenges and opportunities," *J. Netw. Comput. Appl.*, vol. 144, pp. 13–48, Oct. 2019.

[6] M. Andoni *et al.*, "Blockchain technology in the energy sector: A systematic review of challenges and opportunities," *Renewable Sustain. Energy Rev.*, vol. 100, pp. 143–174, Feb. 2019.

[7] H.-N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8076–8094, Oct. 2019.

[8] C.-W. Ten, K. Yamashita, Z. Yang, A. V. Vasilakos, and A. Ginter, "Impact assessment of hypothesized cyberattacks on interconnected bulk power systems," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4405–4425, Sep. 2018.

[9] H. Zhang, J. Wang, and Y. Ding, "Blockchain-based decentralized and secure keyless signature scheme for smart grid," *Energy*, vol. 180, pp. 955–967, Aug. 2019.

[10] L. Chen, W. K. Lee, C. C. Chang, K. K. R. Choo, and N. Zhang, "Blockchain based searchable encryption for electronic health record sharing," *Future Gener. Comput. Syst.*, vol. 95, pp. 420–429, Jun. 2019.

[11] T. McGhin, K.-K. R. Choo, C. Z. Liu, and D. He, "Blockchain in healthcare applications: Research challenges and opportunities," *J. Netw. Comput. Appl.*, vol. 135, pp. 62–75, Jun. 2019.

[12] J. Hu, D. He, Q. Zhao, and K.-K. R. Choo, "Parking management: A blockchain-based privacy-preserving system," *IEEE Consum. Electron. Mag.*, vol. 8, no. 4, pp. 45–49, Jul. 2019.

[13] M. Banerjee, J. Lee, and K. K. R. Choo, "A blockchain future for Internet of Things security: A position paper," *Digit. Commun. Netw.*, vol. 4, no. 3, pp. 149–160, Aug. 2018.

[14] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: Perspectives and challenges," *Wireless Netw.*, vol. 20, no. 8, pp. 2481–2501, 2014.

[15] Z. Sheng, S. Yang, Y. Yu, A. V. Vasilakos, J. A. Mccann, and K. K. Leung, "A survey on the IETF protocol suite for the Internet of Things: Standards, challenges, and opportunities," *IEEE Wireless Commun.*, vol. 20, no. 6, pp. 91–98, Dec. 2013.

[16] M. Wazid, A. K. Das, N. Kumar, and A. V. Vasilakos, "Design of secure key management and user authentication scheme for fog computing services," *Future Gener. Comput. Syst.*, vol. 91, pp. 475–492, Feb. 2019.

[17] M. Wazid, A. K. Das, K. VivekanandaBhat, and A. V. Vasilakos, "LAM-CIoT: Lightweight authentication mechanism in cloud-based IoT environment," *J. Netw. Comput. Appl.*, vol. 150, Jan. 2020, Art. no. 102496.

[18] 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," *J. Netw. Comput. Appl.*, vol. 116, pp. 42–52, Aug. 2018.

[19] R. Chaudhary, A. Jindal, G. S. Aujla, S. Aggarwal, N. Kumar, and K.-K. R. Choo, "BEST: Blockchain-based secure energy trading in SDN-enabled intelligent transportation system," *Comput. Security*, vol. 85, pp. 288–299, Aug. 2019.

[20] A. S. Musleh, G. Yao, and S. M. Muyeen, "Blockchain applications in smart grid-review and frameworks," *IEEE Access*, vol. 7, pp. 86746–86757, 2019.

[21] S. Nakamoto. 2008. *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://downloads.coindesk.com/research/whitepapers/bitcoin.pdf

[22] P. Danzi, A. E. Kalor, C. Stefanovic, and P. Popovski, "Delay and communication tradeoffs for blockchain systems with lightweight IoT clients," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2354–2365, Apr. 2019.

[23] Z.-Y. Ai, Y.-T. Zhou, and F. Song, "A smart collaborative routing protocol for reliable data diffusion in IoT scenarios," *Sensors*, vol. 18, no. 6, p. 1926, 2018.

[24] F. Song, M. Zhu, Y. Zhou, I. You, and H. Zhang, "Smart collaborative tracking for ubiquitous power IoT in edge-cloud interplay domain," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6046–6055, Jul. 2020.

[25] I. A. Kamil and S. O. Ogundoyin, "EPDAS: Efficient privacy-preserving data analysis scheme for smart grid network," *J. King*

Saud Univ. Comput. Inf. Sci., to be published. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1319157818308085

[26] J.-L. Tsai and N.-W. Lo, "Secure anonymous key distribution scheme for smart grid," *IEEE Trans. Smart Grid*, vol. 7, no. 2, pp. 906–914, Mar. 2016.

[27] W. Su and A. Q. Huang, *The Energy Internet*. Sawston, U.K.: Woodhead Publ., 2018.

[28] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, 2002.

[29] R. Canetti and H. Krawczyk, "Universally composable notions of key exchange and secure channels," in *Int. Conf. Theory Appl. Cryptogr. Tech.*, Amsterdam, The Netherlands, 2002, pp. 337–351.

[30] AVISPA. *Automated Validation of Internet Security Protocols and Applications*. Accessed: Oct. 2019. [Online]. Available: http://www.avispa-project.org/

[31] *MIRACL Cryptographic SDK: Multiprecision Integer and Rational Arithmetic Cryptographic Library*. Accessed: Apr. 2020. [Online]. Available: https://github.com/miracl/MIRACL

[32] S.-K. Kim and J.-H. Huh, "A study on the improvement of smart grid security performance and blockchain smart grid perspective," *Energies*, vol. 11, no. 8, pp. 1–22, 2018.

[33] N. Wang *et al.*, "When energy trading meets blockchain in electrical power system: The state of the art," *Appl. Sci.*, vol. 9, no. 8, p. 1561, 2019.

[34] A. C.-F. Chan and J. Zhou, "Cyber-physical device authentication for the smart grid electric vehicle ecosystem," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 7, pp. 1509–1517, Jul. 2014.

[35] T. W. Chim, S. M. Yiu, V. O. K. Li, L. C. K. Hui, and J. Zhong, "PRGA: Privacy-preserving recording & gateway-assisted authentication of power usage information for smart grid," *IEEE Trans. Depend. Secure Comput.*, vol. 12, no. 1, pp. 85–97, Jan./Feb. 2015.

[36] D. He, H. Wang, M. K. Khan, and L. Wang, "Lightweight anonymous key distribution scheme for smart grid using elliptic curve cryptography," *IET Commun.*, vol. 10, no. 14, pp. 1795–1802, Sep. 2016.

[37] H. J. Jo, I. S. Kim, and D. H. Lee, "Efficient and privacy-preserving metering protocols for smart grid systems," *IEEE Trans. Smart Grid*, vol. 7, no. 3, pp. 1732–1742, May 2016.

[38] H. Li, R. Lu, L. Zhou, B. Yang, and X. Shen, "An efficient merkle-tree-based authentication scheme for smart grid," *IEEE Syst. J.*, vol. 8, no. 2, pp. 655–663, Jun. 2014.

[39] H. Nicanfar, P. Jokar, K. Beznosov, and V. C. M. Leung, "Efficient authentication and key management mechanisms for smart grid communications," *IEEE Syst. J.*, vol. 8, no. 2, pp. 629–640, Jun. 2014.

[40] V. Odelu, A. K. Das, M. Wazid, and M. Conti, "Provably secure authenticated key agreement scheme for smart grid," *IEEE Trans. Smart Grid*, vol. 9, no. 3, pp. 1900–1910, May 2018.

[41] N. Saxena, B. J. Choi, and R. Lu, "Authentication and authorization scheme for various user roles and devices in smart grid," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 5, pp. 907–921, May 2016.

[42] Y. Zhou, Y. Guan, Z. Zhang, and F. Li, "A blockchain-based access control scheme for smart grids," in *Proc. IACR Cryptol. ePrint Archive*, 2019, p. 880. [Online]. Available: https://eprint.iacr.org/2019/880.pdf

[43] M. Qi and J. Chen, "Two-pass privacy preserving authenticated key agreement scheme for smart grid," *IEEE Syst. J.*, early access, May 19, 2020, doi: 10.1109/JSYST.2020.2991174.

[44] S. A. Chaudhry, H. Alhakami, A. Baz, and F. Al-Turjman, "Securing demand response management: A certificate-based access control in smart grid edge computing infrastructure," *IEEE Access*, vol. 8, pp. 101235–101243, 2020.

[45] K. Mahmood, S. A. Chaudhry, H. Naqvi, S. Kumari, X. Li, and A. K. Sangaiah, "An elliptic curve cryptography based lightweight authentication scheme for smart grid communication," *Future Gener. Comput. Syst.*, vol. 81, pp. 557–565, Apr. 2018.

[46] D. Abbasinezhad-Mood and M. Nikooghadam, "Design and hardware implementation of a security-enhanced elliptic curve cryptography based lightweight authentication scheme for smart grid communications," *Future Gener. Comput. Syst.*, vol. 84, pp. 47–57, Jul. 2018.

[47] K. Mahmood *et al.*, "Pairing based anonymous and secure key agreement protocol for smart grid edge computing infrastructure," *Future Gener. Comput. Syst.*, vol. 88, pp. 491–500, Nov. 2018.

[48] J. Wang, L. Wu, K.-K. R. Choo, and D. He, "Blockchain-based anonymous authentication with key management for smart grid edge computing infrastructure," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1984–1992, Mar. 2020.

[49] K. Gai, Y. Wu, L. Zhu, L. Xu, and Y. Zhang, "Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7992–8004, Oct. 2019.

[50] H. Yao, T. Mai, J. Wang, Z. Ji, C. Jiang, and Y. Qian, "Resource trading in blockchain-based industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3602–3609, Jun. 2019.

[51] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. 29, no. 2, pp. 198–208, Mar. 1983.

[52] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *IEEE Trans. Comput.*, vol. 51, no. 5, pp. 541–552, May 2002.

[53] S. Challa *et al.*, "Secure signature-based authenticated key establishment scheme for future IoT applications," *IEEE Access*, vol. 5, pp. 3028–3043, 2017.

[54] C.-C. Chang and H.-D. Le, "A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 357–366, Jan. 2016.

[55] 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 Things J.*, vol. 5, no. 6, pp. 4900–4913, Dec. 2018.

[56] S. Malani, J. Srinivas, A. K. Das, K. Srinathan, and M. Jo, "Certificate-based anonymous device access control scheme for IoT environment," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9762–9773, Dec. 2019.

[57] M. Wazid, A. K. Das, V. Odelu, N. Kumar, M. Conti, and M. Jo, "Design of secure user authenticated key management protocol for generic IoT networks," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 269–282, Feb. 2018.

[58] A. K. Das, S. Zeadally, and D. He, "Taxonomy and analysis of security protocols for Internet of Things," *Future Gener. Comput. Syst.*, vol. 89, pp. 110–125, Dec. 2018.

[59] W. E. May, "Secure hash standard," Nat. Inst. Stand. Technol., U.S. Dept. Commerce, Gaithersburg, MD, USA, Rep. FIPS PUB 180-4, Apr. 1995. Accessed: Aug. 2019. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

[60] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Security*, vol. 1, no. 1, pp. 36–63, 2001.

[61] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly secure key distribution for dynamic conferences," *Inf. Comput.*, vol. 146, no. 1, pp. 1–23, 1998.

[62] M. Abdalla, P.-A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Proc. 8th Int. Workshop Theory Pract. Public Key Cryptogr. Lecture Notes Comput. Sci.*, 2005, pp. 65–84.

[63] J. Srinivas, A. K. Das, N. Kumar, and J. J. P. C. Rodrigues, "TCALAS: Temporal credential-based anonymous lightweight authentication scheme for Internet of drones environment," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6903–6916, Jul. 2019.

[64] B. Bera, S. Saha, A. K. Das, N. Kumar, P. Lorenz, and M. Alazab, "Blockchain-envisioned secure data delivery and collection scheme for 5G-based IoT-enabled Internet of drones environment," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9097–9111, Aug. 2020, doi: 10.1109/TVT.2020.3000576.

[65] B. Bera, D. Chattaraj, and A. K. Das, "Designing secure blockchain-based access control scheme in IoT-enabled Internet of drones deployment," *Comput. Commun.*, vol. 153, pp. 229–249, Mar. 2020.

[66] 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 Things J.*, vol. 7, no. 4, pp. 3184–3197, Apr. 2020.

[67] AVISPA. *SPAN, The Security Protocol Animator for AVISPA*. Accessed: Oct. 2019. [Online]. Available: http://www.avispa-project.org/

[68] *Raspberry Pi 3 Model B+*. Accessed: Jun. 2020. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/

[69] K. Khullar. *Implementing PBFT in Blockchain*. Accessed: Jul. 2020. [Online]. Available: https://medium.com/coinmonks/implementing-pbft-in-blockchain-12368c6c9548

Basudeb Bera (Student Member, IEEE) received the M.Sc. degree in mathematics and computing from Indian Institute of Technology (Indian School of Mines) Dhanbad, Dhanbad, India, in 2014, and the M.Tech. degree in computer science and data processing from Indian Institute of Technology Kharagpur, Kharagpur, India, in 2017. He is currently pursuing the Ph.D. degree in computer science and engineering from the Center for Security, Theory and Algorithmic Research, Indian Institute of Technology Hyderabad, Hyderabad, India.

His research interests are cryptography, network security, and blockchain technology. He has published five papers in international journals and conferences in the above areas.

Sourav Saha (Student Member, IEEE) received the Bachelor of Technology (B.Tech.) degree in computer science and engineering from the Central Institute of Technology Kokrajhar, Kokrajhar, India, and the Master of Science (M.S.) by Research degree in computer science and engineering from Indian Institute of Information Technology, Sri City, India. He is currently pursuing the Ph.D. degree in computer science and engineering with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology Hyderabad, Hyderabad, India.

His research interests include network security and blockchain technology. He has published three papers in international journals and conferences in the above areas.

Ashok Kumar Das (Senior Member, IEEE) received the M.Tech. degree in computer science and data processing, the M.Sc. degree in mathematics, and the Ph.D. degree in computer science and engineering from the Indian Institute of Technology Kharagpur (IIT Kharagpur), Kharagpur, India.

He is currently an Associate Professor with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology Hyderabad, Hyderabad, India. His current research interests include cryptography and network security, including security in smart grid, Internet of Things, Internet of Drones, Internet of Vehicles, cyber–physical systems and cloud computing, blockchain, and AI/ML security. He has authored over 235 papers in international journals and conferences in the above areas, including over 200 reputed journal papers.

Prof. Das was a recipient of the Institute Silver Medal from IIT Kharagpur. He is on the editorial board of IEEE SYSTEMS JOURNAL, *Computer Communications* (Elsevier), *IET Communications*, *KSII Transactions on Internet and Information Systems*, and *International Journal of Internet Technology and Secured Transactions* (Inderscience). He is a Guest Editor for *Computers and Electrical Engineering* (Elsevier) for the special issue on Big data and IoT in e-healthcare and for *ICT Express* (Elsevier) for the special issue on Blockchain Technologies and Applications for 5G Enabled IoT, and has served as a Program Committee Member in many international conferences.

Athanasios V. Vasilakos (Senior Member, IEEE) received the Ph.D. degree in computer engineering from the University of Patras, Patras, Greece, in 1988.

He is with the School of Electrical and Data Engineering, University Technology Sydney, Sydney, NSW, Australia, the Department of Computer Science and Technology, Fuzhou University, Fuzhou, China, and the Department of Computer Science, Electrical and Space Engineering, and Lulea University of Technology, Lulea, Sweden. He has published over 700 technical research papers in leading journals and conferences in his areas of research.

Dr. Vasilakos is Web of Science 2017, 2018, 2019, and 2020 Highly Cited Researcher. He served or is serving as an editor for many technical journals, such as the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, IEEE TRANSACTIONS ON CLOUD COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON NANOBIOSCIENCE, IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE, IEEE TRANSACTIONS ON CLOUD COMPUTING, *IEEE Communication Magazine*, *ACM Transactions on Autonomous and Adaptive Systems*, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, and *ACM Transactions on Autonomous and Adaptive Systems*.