Peer-to-Peer Networking and Applications
https://doi.org/10.1007/s12083-024-01709-8

Consortium blockchain based secure and efficient data aggregation and dynamic billing system in smart grid

Ozgur Oksuz[1]

Received: 27 March 2023 / Accepted: 6 April 2024

Abstract
In a smart grid, collected electricity consumption periodically from smart meters allow entities to bill the customers, power company to operate the grid successfully, and users to control the use of their appliances. However, energy consumptions of users should be protected since the data provides the user's daily habit that an adversary uses the data to extract useful information about the users. Moreover, users' identities should not be disclosed to untrusted entities since the untrusted entities map identities to their real identities. In this paper, we propose a system that protects users' data privacy using multi-pseudorandom identities and a randomization technique. Moreover, the proposed work provides fast authentication for smart meters to send their readings to data aggregators. Furthermore, the proposed work is based on consortium blockchain to eliminate a single point of failure and provides transparency of messages and operations. In addition, we use dynamic billing and pricing mechanism for the users to see their bills.

Keywords Anonymity · Unlinkability · Blockchain · PBFT · Dynamic billing · Dynamic pricing

1 Introduction

Smart grid allows entities, Smart Meters (*SM*), Data Aggregators (*DA*), Control Center (*CC*), Bill Provider (BP) to work together to make grid stable, reliable and secure. The stability and reliability of the grid depend on a balance in energy production and consumption within an electrical grid. In other words, the energy generated must equal the energy consumed for a region. If the generated energy is less than the consumed energy in a region, the households in the region face blackouts. To stable the power grid, *CC* needs to have users' consumption data periodically to analyze the consumed energy for that region. Having periodic consumption data allows *CC* to figure out how much energy needs to be produced for that region so that the households can continue their lives without outages. Users consumption data are sent to *CC* via *SM*s located in the users' home. A *SM* collects energy usage periodically (every 15 or 30 mins) and sends it to *CC* via a gateway (*DA*). However, collected energy usage for each period leaks users' daily life. In other words, users' data privacy is compromised by untrusted parties. A solution for protecting users' data privacy is to use encryption. *SM*s encrypt readings before they send to *CC*. However, *SM*s have limited computation resources. Another method is using anonymization. *SM*s use pseudo anonymous identities not their real identities once they send periodic readings. However, using a pseudo anonymous identity once they send readings results in another privacy problem. Even if each reading is anonymously sent to *CC*, an adversary links behavior pattern of the user. In other words, the connection between user's pseudonyms may be disclosed by matching the electricity consumption profile with user behaviors in a particular periods of time. The adversary figures out that two anonymous profiles belong to the same user [1, 2].

Using pseudo-anonymous identities for *SM*s provides some other disadvantages. The first disadvantage is when a user tries to tamper their corresponding *SM* not to send a periodic reading to decrease their bills, or a software problem happens so that the corresponding *SM* does not send data to the *DA*. It results load unbalancing in that region and incorrect billing. Tempered *SM*s need to be found and

This article is part of the Topical Collection: *Special Issue on 2 - Track on Security and Privacy*
Guest Editor: Rongxing Lu

✉ Ozgur Oksuz
ooksuz@ktun.edu.tr

[1] Department of Software Engineering, Konya Technical University, Konya 42250, Turkey

eliminated in the grid immediately. However, using pseudo anonymous identities for the *SM*s make this difficult. The second one is to authenticate each *SM*'s identity when they send readings to *CC* via a *DA*. However, this increases communication costs between each *SM* and the trusted party linearly in the number of identities. The third disadvantage happens when the entity bills users. Since *SM*s do not use their real identities, these pseudo anonymous identities should be linked. The corresponding usages of the users should be aggregated. Then, each user needs to be billed accordingly. Lastly, the collected data should not be stored in a central database that results single point of failure. This data should be protected to internal and external attacks.

To address these problems above, we propose a secure data aggregation and dynamic billing system based on consortium blockchain in smart grid. The Integration of blockchain and smart grid result in many advantages. One of the advantages is transparency and availability of data. In the blockchain, each user's power consumption data is stored periodically by a data aggregator. Moreover, the price of a period can be also stored in the blockchain. Once the data is stored in the blockchain, users easily track their data and bills. They can verify them. Thus, each user can verify its power consumption data and bills any time. Another advantage is to provide decentralized trust. Blockchain is decentralized system that consists of nodes (computers). Each node has a full copy of the blockchain. In other words, each node has the blocks that contain transactions. The nodes should agree on which block should be added to the blockchain using consensus. The nodes do not trust each other. If one of the nodes stop working, the data is still available from other nodes. This results in that the blockchain eliminates single point of failure attacks. Moreover, it is infeasible to tamper any data in a block. The blocks are connected to each other by cryptography. If there is a change in a block, this change is going to be discarded by majority of honest nodes. However, in a centralized system, the database is controlled by a single entity that can be hacked or tampered easily.

Our contributions are as follows:

- The proposed system provides fast authentication mechanism for data aggregators to ensure received transactions issued from real smart meters. This decreases communication cost between each *SM* and *KGCBP* from $O(n)$ to $O(1)$, where n is the number of identities of each *SM*.
- Entities figure out if there are faulty smart meters (tempered or software/hardware issues). Then, the entities find out and inform the trusted party to eliminate tempered smart meters from the grid or punish the corresponding consumers. Finding faulty smart meters eliminates the unbalancing the load in the grid and incorrect billing for the users.
- Our system uses multiple pseudo random identities instead of single pseudo random identities or real identities and a randomization technique to randomize smart meters' real data to provide data privacy of the consumers. This provides unlinking the behavior patterns of the users.
- The system provide dynamic billing system for the consumers to see their bill privately.
- It uses consortium blockchain to eliminate single point of failure for data aggregation, dynamic billing and pricing. The proposed system uses consortium blockchain that includes voting based Practical Byzantine Fault Tolerance *vPBFT* consensus mechanism to entities agree transactions (messages) before they are put in the blockchain.
- We also give concrete proof of the security of the proposed system.

Paper organization The rest of the paper is as follows: Section 2 presents related work. Section 3 gives some definitions used throughout the paper. Section 4 introduces the proposed architecture and threat model of the proposed system. Section 5 gives the proposed system. Section 6 represents the security analysis of the system. Section 7 gives some discussions about the proposed system. Section 8 concludes the paper and provides possible future work.

2 Related work

The study in [3] proposed an anonymization technique to protect users' data. In their work, 2 different identities are assigned to a smart meter. One identity (not anonymous) is used when the smart meter sends low frequency power consumption (monthly) to the utility for billing purposes. The other one (anonymous) is used when it sends high frequency power consumption data (for every 15 or 30 mins.) to the utility for operational reasons. However, the work in [1, 2] showed that using a monthly consumption data of a user (low frequency data) and anonymized high frequency dataset, an adversary immediately de-anonymized the smart metering data. A method for de-anonymizing the data is taking the sums of half-hourly values will match this aggregate. Moreover, the work in [3] is not based on blockchain and does not consider dynamic billing.

The work in [4] proposed a privacy preserving aggregation and dynamic billing scheme based on private blockchain that uses bloom filter data structure and zero knowledge proof system for fast authentication while protecting user's privacy using the multi pseudo identities for each user. Before sending power consumption data to control center, each user uses pseudo identities not their real identities. These pseudo identities are obtained from key generation center (*KGC*). However, in this work, each smart meter needs to agree on a mining node (a smart meter) to issue transaction into blockchain. For each period, each smart meter needs to send $O(m)$ information to all smart meters in the same region and checks $O(m)$ identities in bloom filter and signatures for verifying origin of the message. Moreover, in this protocol, each smart meter needs to store linear size pseudo identities to generate transactions. Furthermore, key generation center sends $O(n)$ identities to each smart meter. The protocol in [4] does not have any consensus algorithm.

Fan and Zhang [5] proposes a data aggregation and monitoring mechanism based on consortium blockchain. This scheme aggregates multi-dimensional data (power consumption, temperature, humidity, etc.) and provides feedback from multiple receivers, but the scheme uses bilinear pairing operations, which results in high computational overhead. Moreover, the system uses encryption to allow each receiver to obtain their corresponding data by decryption. For example, the smart meter encrypts the multidimensional data for each individual receiver (control center, grid operator and supplier). This provides also additional computational cost for the smart meter. Moreover, each smart meter uses the same identity once it sends consumption data to untrusted entity, an untrusted receiver in the blockchain can figure out the smart meters all consumptions. They can be able to figure out daily life behavior of the user even though the physical address of the smart meter is not known by them. In our work, we hide the smart meter's identity each time. Moreover, we use multi pseudo identities for each smart meter to hide power usage of each smart meter.

Wu et al. [6] proposed a new transmission scheme for a smart grid among the smart meter (*SM*), the electricity utility (*EU*), and the trusted authority (*TA*). Each *SM* sends power consumption data to specific *EU* anonymously. If the *SM*'s power consumption data is above or below the threshold, the *EU* is then forward the data to the trusted party to expose the identity of the *SM* for incentive (reward or punishment). In their system, bilinear pairing cryptography and *SM*s perform bunch of computations. Moreover, this work does not consider dynamic pricing and billing. Furthermore, it is not based on blockchain.

Zhang et al. [7] proposed a consortium based blockchain in smart grid. The system uses aggregated ring signcryption to achieve anonymity of a *SM*'s power consumption before the smart meter sends its actual power consumption. In their protocol, each smart meter is assigned a pseudonym identity. Then this pseudo identity is appended to actual power consumption value to send to assign community area gateway (*BG*) using certificateless ring signcryption. Once the *BG* and control center (*CC*) receives the message from the smart meter, they can be able to retrieve actual consumption and pseudo identity of the smart meter. Since the same smart meter uses the same pseudo identity, the *BG* and *CC* can be able to figure out the smart meters all consumptions. Moreover, they can be able to figure out daily life behavior of the user even though the physical address of the smart meter is not known by them. In our work, any unauthorized entity is not able to find out individual user's consumption data. Our work hides the smart meter's identity each time. Moreover, we use multi pseudo identities for each smart meter.

In work [8], authors proposed a framework that protects users' power consumption data from honest but curious adversary. Each smart meter sends its power consumption data to n aggregators that at most n-1 aggregators are semi honest adversary using Shamir's secret sharing framework. Then, each aggregator sums up all the values receives from smart meters to have the aggregated consumption value and forwards to the utility for grid functionalities. Thus, neither aggregators nor the utility does not know any specific smart meter's power consumption value. However, this protocol does not examine dynamic billing system for each user. Moreover, it is not based on blockchain.

The study in [9] proposed a privacy preserving data aggregation scheme based on blockchain. This system uses Paillier encryption and *BLS* signatures to protect users' privacy and integrity of the messages. A leader smart meter is chosen among all smart meters in an area/region to aggregate users' power consumption data and put transactions to blockchain. In the system, most of the work is done by smart meters so this results scalability problem since smart meters have limited resource capabilities (computation). Moreover, the work in [9] only focuses on privacy preserving data aggregation of users'. Furthermore, it does not consider dynamic pricing and billing.

The study in [10] proposed a blockchain based data aggregation and dynamic billing system that uses Schnorr signature to provide message integrity and pseudo random identities to hide users' consumption values. However, in [10], only the single entity a trusted party (control center) is allowed to issue transactions into blockchain. Each smart meter sends their power consumption data to control center. The control center checks if the sender's authenticity. Then, it adds the transaction to the ledger. Since only single entity issue transactions into blockchain, it has single point of failure problem. If this entity is not available, the system does not work properly. Moreover, there is no consensus mechanism. Furthermore, it does not consider to find faulty *SM*s to prevent unbalancing the grid.

The study in [11] uses Schnorr signatures to provide data integrity and uses an onion routing network (*TOR*) to ensure anonymous communication between smart meters and control center. However, once a smart meter sends its consumption value to control center, it needs to encrypt the data with bunch of public keys to provide anonymity of the user. In this work, the control center is the only entity that writes transactions into blockchain. Thus, it has single point of failure problem. The scheme in [11] has scalability problem since each smart meter sends its multiple encrypted data to control center that decrypts and put the power consumption data into blockchain. If the control center is not available, the whole system does not work properly. Moreover, each smart meter has limited computational operation, using *TOR* network is not scale well. Furthermore, it does not consider to find faulty *SM*s to prevent unbalancing the grid.

The work in [12] proposed a blockchain based secure data aggregation and distributed power dispatching for microgrids. Their protocol uses multi-round communication protocol to establish a key pair between a smart meter and service provider. This pairwise key establishment phase is done for each transaction that the smart meter sends to the service provider. Using this symmetric key, the smart meter encrypts its reading and send to the service provider with its public key and signature. The service provider than decrypts the received message and re-encrypts with its public key and put the transaction into the transaction pool. The transactions in the pool consist of smart meters public keys that allows an adversary (service provider) to link transactions. In this system, the entities, smart meter and service provider, and service provider and service provider establish pairwise keys for each communication. These keys are used in voting based *PBFT* consensus to add the transactions to the blockchain. Moreover, this system does not consider dynamic billing feature in their paper.

Abouyoussef and Ismail [13] proposed a blockchain based privacy preserving demand response management (DMS). In this system, users' anonymity is preserved using random identifiers and group signature. Moreover, the system performs customer billing and auditing of suspicious customers if needed. However, their adversarial model is limited only for honest but curious adversary. In addition, for achieving anonymity each smart meter uses different random number for its identifier that needs to be remembered. Thus, results that the storage size of a user is linear in the size of random numbers. In addition, even the smart meter identity is anonymized for each actual power consumption to send to utility, an adversary can be able to map the smart meter's random identities to a specific user. This happens when a user has the same daily routine. For example, the user wakes up at 7, and uses hair dryer at 7:30 and uses kettle 8:00 am on weekdays. In this case, the power consumptions of the user are the same on weekdays.

The work in [14] proposed decentralized privacy reserving aggregation scheme that uses encryption to provide confidentiality of consumption data and signature to provide integrity of the messages between entities. Their work uses paring based cryptography for verification of messages resulting heavy computational cost for the users. Moreover, the work in [14], does not consider dynamic billing feature in their system.

The study in [15], the authors proposed blockchain based data aggregation system that smart meters send encrypted electricity consumption data and signature of the encrypted data to an aggregator. The aggregator aggregates the ciphertexts (homomorphic encryption) and executes *PBFT* consensus algorithm. Then it sends to corresponding book keeping node responsible to add the transactions to the blockchain. Since the individual reading is encrypted and the aggregated readings are available in the cleartext, the users data privacy is preserved. However, the work in [15] does not consider dynamic billing system for the users.

The study in [16], proposed a blockchain based framework for privacy preserving verifiable billing in smart meter. In their scheme, each smart meter uses homomorphic encryption for not leaking actual power consumption of its corresponding user. It uses encryption and signature algorithms to provide verification of smart meters' bills. Their scheme uses pairing based cryptography which results heavy computations. However, the price of electricity is not changed a year. Thus, their scheme does not include dynamic pricing property. Moreover, the scheme does not provide any consensus mechanism. Furthermore, community gateway (*CG*) which aggregates power consumption of smart meters is not malicious adversary. Malicious behavior of CG can easily change received encrypted power consumption data when it issues transaction in the blockchain. Incorrect power consumption data results load balancing problem in the grid and incorrect billing for the corresponding smart meter/user. Our protocol provides dynamic pricing. Moreover, it provides a consensus algorithm before a transaction is added to blockchain. Furthermore, our protocol is resilient to malicious adversaries. In addition, our scheme provides lightweight computational operations.

The study in [17] proposed a lightweight certificateless aggregate ring signature scheme for privacy protection in smart grids. In this scheme, certificateless cryptosystem is used to avoid key escrow and certificates management problems and ring signature is used to ensure the unconditional anonymity of users. However, the authors did not examine how the users were billing.

The authors in [18] proposed an efficient and robust multi-dimensional data aggregation scheme based on blockchain. In this work, smart meters in a region select a leader (mining node) to aggregate consumption values and add transactions to blockchain. For this selection, Raft protocol is used. Pedersen secret sharing scheme is used to protect data privacy and integrity of messages. Each user sends $O(t)$ commitment values to other smart meters. The communication complexity between a smart meter is $O(mt)$ (m smart meters). Storage complexity of a smart meter is $O(mt)$ group element in G. Then each smart meter sends $O(m)$ shares to other smart meters. Moreover, each smart meter does $O(m)$ verifications.

In Table 1, the proposed work is compared with other studies based on the features: Multiple Writers, Unlinkability

Table 1 Feature Comparisons of Blockchain Based Studies with the Proposed Work

Reference	Multiple Writers	Unlinkability and Anonymity	Dynamic Billing	Consensus	Method
[4]	Yes	Yes	Yes	No	PRI
[5]	Yes	No	No	DPOS	Encryption
[7]	Yes	No	Yes	PBFT	Signcryption
[9]	Yes	No	No	No	Encryption
[10]	No	Yes	Yes	No	Key Agreement + PRI
[11]	No	Yes	Yes	No	Encryption + PRI
[12]	Yes	No	No	vPBFT	Encryption
[15]	Yes	No	No	PBFT	Encryption
[14]	Yes	No	No	PBFT	Encryption
[13]	Yes	No	Yes	No	RI
[16]	Yes	No	No	No	Encryption
[18]	Yes	No	No	Raft	Secret Sharing
This work	Yes	Yes	Yes	vPBFT	PRI

and Anonymity, Consensus Mechanism, Dynamic Billing and Method. Multiple-writers property provides multiple entities to write transactions to the blockchain. In other words, multiple entities (nodes) write transactions and create blocks in the system. Moreover, each node stores a full copy of the blocks. This property results in eliminating single point-of-failure attacks. Unlinkability and Anonymity property means that multiple transactions of the same smart meter are not linked (unlinkability), and for each transaction smart meters are not identified (anonymity). Consensus shows if the system has a mechanism to verify the transactions before they are put into the blockchain. Dynamic Billing describes if the system has dynamic pricing and billing mechanism. Method feature provides how the data privacy of the users once smart meters send power consumption data to data aggregators is achieved. In the table, *PRI* stands for Pseudo Random Identity, *RI* stands for Random Identity.

In Table 2, the proposed system's computational costs are compared to other studies that do not use encryption once smart meters send power consumption data to data aggregators in their system. In $Computation(SM)$ column indicates a smart meter's computational complexity, Sig is the signature, G group multiplication, SV is the signature verification, V is the verification in G using *Pedersen* secret sharing commitment, H is the hash, and Add is the addition operation in Z_q. Moreover, $Communication(SM - SM)$ column indicates the communication complexity of a smart meter with other smart meters, G is the group element is sent from an SM to other SMs, SS is the secret sharing element (in Z_q) is sent from an SM to other SMs. These costs are the monthly costs for an SM. Furthermore, in the table, m represents the number of smart meters in each area, n represents the number of identities used in a month.

In the proposed system, each smart meter only stores $O(1)$ information. Moreover, the smart meters do not need to communicate each other to form a transaction. Furthermore, the communication complexity between each smart meter and key generation center is $O(1)$. The proposed system has better communication and storage complexities and security properties than other studies in related work. The proposed work satisfies unlinkability and anonymity property that smart meters' multiple transactions are not linkable. The proposed system has a mechanism to find faulty smart meters to prevent load unbalancing and incorrect billing. It eliminates single-point-of-failure attacks and centralization using multiple writes.

Table 2 Storage, Computation, and Communication Comparisons for The Systems that do not use Encryption

Reference	Storage (SM)	Computation (SM)	Communication (SM-SM)	Communication (SM-KGC)	Finding Faulty SMs
[4]	$O(n)$	$O(n)\ Sig + O(mn)\ SV$	$O(mn)\ G + O(mn)\ Sig$	$O(n)$	No
[13]	$O(n)$	$O(n)\ H + O(n)\ Add + O(n)\ Sig$	-	$O(1)$	No
[18]	$O(mt)$	$O(n)\ G + O(n)\ H + O(n)\ Sig + O(m)\ V$	$O(mt)\ G + O(m)\ SS$	$O(1)$	No
This work	$O(1)$	$O(n)\ H + O(n)\ Add + O(n)\ G + O(n)\ Sig$	-	$O(1)$	Yes

3 Definitions

Definition 1 (Pseudo Random Generator). A pseudo-random generator (*PRG*) outputs strings that are computationally indistinguishable from random strings. More precisely, we say that a function $G : \{0,1\}^{\kappa} \rightarrow \{0,1\}^{\xi}$, where $\xi > \kappa$ is a (t, ϵ)-pseudo-random generator if

1. G is efficiently computable by a deterministic algorithm,
2. for all t time probabilistic algorithm $\mathcal{A}$, $|\Pr[\mathcal{A}(G(s)) = 0|s \leftarrow \{0,1\}^{\kappa}] - \Pr[\mathcal{A}(r) = 0|r \leftarrow \{0,1\}^{\xi}]| \leq \epsilon$.

In Definition 1, s is the secret key (seed) chosen uniformly random from $\{0,1\}^{\kappa}$, where κ is the security parameter. r is a random number chosen from $\{0,1\}^{\xi}$, where $\kappa < \xi$. The second property of a *PRG* function says that a polynomial-time adversary can not distinguish an output string of a *PRG* function using a secret $s \in \{0,1\}^{\kappa}$ from a random string r in $\{0,1\}^{\xi}$ with non-negligible probability ϵ.

Definition 2 (Pseudo Random Function). A pseudo-random function (*PRF*) is computationally indistinguishable from a random function - given pairs $\left(x_1, f_s(x_1)\right), \ldots, \left(x_m, f_s(x_m)\right)$, an adversary cannot predict $f_s(x_{m+1})$ for any x_{m+1}. More precisely, we say that a function $f : \{0,1\}^{\kappa} \times \{0,1\}^{\kappa} \rightarrow \{0,1\}^{\kappa}$ is a (t, ϵ, q)-pseudo-random function if

1. $f(s, x) = f_s(x)$ can be computed efficiently from input $x \in \{0,1\}^{\kappa}$ and key $s \in \{0,1\}^{\kappa}$.
2. for any t time oracle algorithm $\mathcal{A}$ that makes at most q adaptive queries, $|\Pr[\mathcal{A}^{f_s(.)} = 0|s \leftarrow \{0,1\}^{\kappa}] - \Pr[\mathcal{A}^{g} = = 0|g \leftarrow F : \{0,1\}^{\kappa} \rightarrow \{0,1\}^{\kappa}]| \leq \epsilon$.

The second property of a *PRF* is that if the adversary issues at most q adaptive queries to an oracle, the adversary can not conclude with non-negligible probability advantage if the oracle is a random function or a pseudo-random function by having those query results. In other words, if the adversary interacts with either a random function or a pseudo-random function with the queries, then the adversary can not tell the difference between the two with non-negligible probability.

Definition 3 (*GGM* − *PRF*). The *GGM*-Based *PRF* [19] is built upon the well-known tree-based *GGM* − *PRF* family. This *PRF* is based on the hierarchical application of any length-doubling *PRG* according to the structure induced by a tree, where input values are uniquely mapped to root-to-leaf paths. Specifically, Let G be a publicly known *PRG* that takes a κ-bit string $s \in \{0,1\}^{\kappa}$ (secret random seed) as input, and outputs a 2κ-bit string, $G(s)$. Let $G_0(s)$ and $G_1(s)$ denote respectively the first and second half of $G(s)$. The GGM-PRF [19] is defined as $F = \{f_s : \{0,1\}^{\kappa} \rightarrow \{0,1\}^{\kappa}\}_{s \in \{0,1\}^{\kappa}}$ such that $f_s\left(x_{\kappa-1} \ldots x_0\right) = G_{x_0}\left(G_{x_1}\left(\cdots\left(G_{x_{\kappa-1}}(k)\right)\right)\right)$, where $x_{\kappa-1} \cdots x_1 x_0$ is the input bitstring of length κ.

GGM − *PRF* forms a binary tree (Fig. 1). A *GGM* is a length doubling function. It takes κ-bit input, and outputs 2κ bits. For κ bit input, there are κ levels on the tree. The root node is labelled as a κ-bit secret seed value s. The children of the root is generated by using *PRG* function. The left child of the root is labelled with the first half of 2κ bits, the right child of the root is labelled with the other half of 2κ bits. The leaves of the tree is the outputs of the pseudo random function (*PRF*) which are indistinguishable from the outputs of an uniformly random function. Definition 3 shows that a random secret results in many pseudo random secrets that are

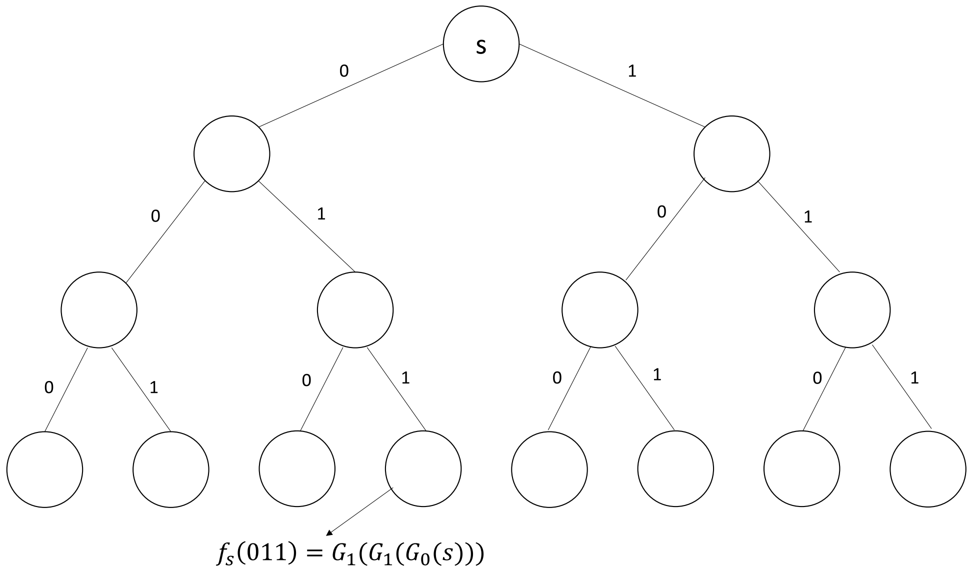

Fig. 1 Illustration of a 3-Level GGM Tree

indistinguishable from totally random values. $GGM - PRF$ lets entities store only one secret to generate multiple secrets. Moreover, it allows entities to share only one secret and generates multiple pseudo random secret keys from it. It decreases the communication and storage complexity between entities. Instead of sharing and storing n keys, entities share only one secret and derive other n keys from it.

Definition 4 (Asymmetric/Public Key Encryption). An asymmetric encryption protocol consists of 3 algorithms: Key generation (*KGen*), Encryption (*Enc*), and Decryption (*Dec*).

KGen algorithm takes a security parameter. It outputs a key pair for user i: Public key Pk_i and Secret key Sk_i. Encryption algorithm takes a message M and public key Pk_i, outputs a ciphertext $C = Enc_{Pk_i}(M)$. Decryption algorithm takes the secret key Sk_i and C, outputs message $M = Dec_{Sk_i}(Enc_{Pk_i}(M))$.

A secure asymmetric key encryption scheme also should be resilient at least to Chosen Plaintext Attack (*CPA*). Examples of *CPA* secure schemes are *RSA* (not the textbook) [20], and Paillier encryption schemes [21].

Definition 5 (Signature). A digital signature protocol consists of 3 algorithms: Key generation (*KGen*), *Sign*, and *Verify*.

KGen algorithm takes a security parameter, outputs a key pair for user i: Signature public key/verification key ($SigPk_i$) and Signature secret key ($SigSk_i$). *Sign* algorithm takes a message M and signature secret key ($SigSk_i$), it outputs a signature $S = Sign_{SigSk_i}(M)$. *Verify* algorithm takes signature public key $SigPk_i$, signature S, and message M, outputs 1 ($Verify(M, S, SigPk_i) == 1$) if the signature is generated by user i under message M. If $Verify(M, S, SigPk_i) == 0$, then the signature under message M is not generated by user i.

A secure signature scheme needs to satisfy two properties: authenticity and integrity. Authenticity says that the owner of the signature convinces a verifier that the owner of the signature generates the signature using the message. Integrity says that signed data cannot be altered by any entity.

An example of secure signature algorithm can be Schnorr signatures [22].

Definition 6 (Blockchain). Blockchain is a $P2P$ decentralized system. It has the ledger technology that it stores transactions up to date. Moreover, the stored data in the ledger is immutable. Each node (computer) keeps a copy of this ledger and checks each transaction's validity. If a transaction is valid, it is added to the ledger. The pile of transactions are stored in the ledger as a block. Blockchain uses two mathematical structures (hash functions and signatures) to provide immutability and integrity of each transaction. There is a special block called the genesis block. The genesis block is the first block. The genesis block does not contain any transaction information in it. Since it is the first block, it has no parent block. The genesis block should be agreed by all the nodes. One node that has the lowest (or highest) address number can create the genesis block and put it into the blockchain.

In blockchain, the peer-to-peer system lets nodes (computers) to keep a full copy of the blockchain and verify the information in the blocks (transactions) to guarantee the data is valid.

Definition 7 (Consortium Blockchain). There are two types of blockchains: Public, and Permissioned (Consortium). In public blockchain, anyone can join, leave, write transactions to the blockchain without any permission from any other parties. The identities are not known by the others. In consortium blockchain, not anyone writes transactions to blockchain. There are nodes are allowed to issue transactions. The identities of the nodes are known by others in the network. The consortium blockchain is defined as there are multiple preselected nodes, the generation of each block is determined by all pre-selected nodes. The consortium blockchain is more flexible and provides better privacy protection compared to the public blockchain. In addition to being used in transactions widely, the consortium blockchain technology can also be used to store data and ensure privacy. Based on the distributed storage characteristics of the consortium blockchain, this paper uses the consortium blockchain to store the power consumption data of the smart home, and the power data is stored in the record pool.

Definition 8 (PBFT Consensus [23]). Practical Byzantine Fault Tolerance (PBFT) consensus mechanism is used to verify the data, the primary/leader/master node is allowed to generate new blocks and broadcasts them to all secondary nodes, the secondary nodes verify the data and feedback the results to the primary node. Once there are enough nodes that agree on the transactions, the leader node adds transactions to the blockchain. In *PBFT* consensus, the system is resilient to f faulty nodes. In order to have a consensus in the system, it should be more than $2f + 1$ non-faulty nodes.

Definition 9 (InterPlanetary File System (*IPFS*)). *IPFS* is a P2P mechanism utilized for keeping records such as images, videos, pdfs, etc. Instead of using location-based addressing, *IPFS* supports content-based addressing. Each file uploaded to *IPFS* returns a unique hash value or Content-Identifier (*CID*), which is used to retrieve the file later. Hash or *CID* is points to the file calculated using the contents of a file. For security, in the proposed system, the encrypted monthly bills of users are stored in *IPFS* and their corresponding hash values in the consortium blockchain. Thus, only authorized users can access the confidential documents. Further, keeping an entire bills of the users on the blockchain is a costly process that affects the overall system's performance since each node has only limited source of computation and storage. Storing the actual file on IPFS and the corresponding hash value of the file inside the consortium blockchain makes the system more efficient and productive.

Definition 10 (Bloom Filter). A *Bloom* filter is data structure that represents a set of $A = \{a_1, \dots, a_\alpha\}$ of α elements and its represented by an array of m bits initially set to 0. The filter uses r independent hash functions $\{HB_1, \dots, HB_r\}$, where $HB_i : \{0,1\}^* \rightarrow [1, m]$ for $i \in [1, r]$. For each item a_i in A, where $i \in \{1, \dots, \alpha\}$, the array bits at positions $HB_1(a_i), \dots, HB_r(a_i)$ are set to 1. A note that any location can be set to 1 multiple times, but only the first is noted. There is, however, some probability of a false positive, in which a_i appears to be in A but actually is not. False positives occur because each location may have also been set by some element other than a_i.

4 Proposed architecture and threat model

In this section, we give the architecture of the proposed system and its workflow. The proposed system has four main actors: Smart Meters (*SM*), Data Aggregators (*DA*), Key Generation Center and Bill Provider (*KGCBP*), and Control Center (*CC*).

Smart Meter (SM): *SM*s are responsible to collect electricity consumption data from home devices (washing machine, kettle, television, computer, refrigerator, etc.) and send it to a data aggregator (*DA*). Smart meters send the users' consumption data as transactions to a data aggregator periodically (for 15 or 30 mins).

Data Aggregator (DA): *DA*s aggregate the users' transactions once the transactions are verified by them. Once the transactions come from the real *SM*s and all checks are successfully done, it executes consensus algorithm with other *DA*s. Once the consensus phase is finished, the corresponding *DA* (leader) adds the transactions to the blockchain.

Control Center (CC): *CC* is responsible for balancing generated and consumed power of a region. Moreover, it is responsible for the calculation of each period's dynamic price. It then puts these prices into the blockchain with the help of *DA*s using consensus.

Key Generation Center and Bill Provider (KGCBP): *KGCBP* initializes system parameters and registers users to the blockchain network. Moreover, it sets public/private keys for smart meters and issues bills as reports for the users. These bill reports consists of users' monthly usage, bill, graphically explained usage for each period. Moreover, it can contain feedback such as an incentive or penalty that the user's consumed energy is less or higher than the threshold electricity consumption. Since the reports take up a lot of space about the users, *KGCBP* sends these reports to *IPFS*.

IPFS: *IPFS* is Inter Planetary File System that is peer-to-peer distributed off-chain storage structure that stores large volumes of files. Once a report is stored in *IPFS*, the corresponding content identifier is returned back to the uploader (*KGCBP*), then these identifier and hash of the file are then put into the blockchain by the help of *DA*s using consensus.

Consortium Blockchain: Consortium blockchain is used to store user's data. Before storing users' energy consumption data, all *DA*s are assigned as the pre-selected nodes of consortium blockchain to participate in consensus. A leader/master/primary node that has a good hardware facilities and strong computing power is chosen with voting mechanism by all other nodes (*DA*s). The primary node is allowed to generate new blocks, and other *DA*s participate in consensus as secondary nodes, rather than the whole network nodes, which reduces the network overhead greatly. The validated data will be permanently stored in the consortium blockchain for user queries.

In the proposed work, with the help of the blockchain, each user can easily verify its bill and usage. The bill and usages are stored at the blockchain and the *IPFS*. The user is able to retrieve this information to compute its bill and usage. Moreover, the blockchain eliminates the central party from the system. This results in that the system is resilient to any single point of failure attacks.

The architecture of the proposed system is illustrated in Fig. 2. The workflow of the proposed system is as follows:

0. *KGCBP* sets up each *SM* by assigning identities to communicate with *DA*s. One identity is for billing, and the other one is for anonymity purposes. Moreover, it sets public/private key pair for each *SM*.
1. Each user (*U*) is registered to the blockchain network by *KGCBP*.
2. Each *SM* sends the electricity consumption data as a transaction to the data aggregator (*DA*).
3. Each *DA* checks the authenticity of the senders, then it aggregates the transactions received from other *SM*s. The aggregated transactions are then included in the consensus process before this is added to the blockchain.
4. Once the consensus process is successfully done, the aggregated transactions are added to the blockchain by the *DA*. It is the leader node that has a good hardware facilities and strong computing power.
5. *CC* retrieves consumption data and analyzes the data for dynamic prices. It issues dynamic pricing for each period. These prices are added to the blockchain by a *DA* that executes consensus.
6. *KGCBP* retrieves prices of each period then issue a monthly bill for each user. Since this bill is issued in detail (daily consumption data, price for each period, etc.), the actual bill is stored in *IPFS* and only the pointer (*CID*) which is the index to the bill is stored

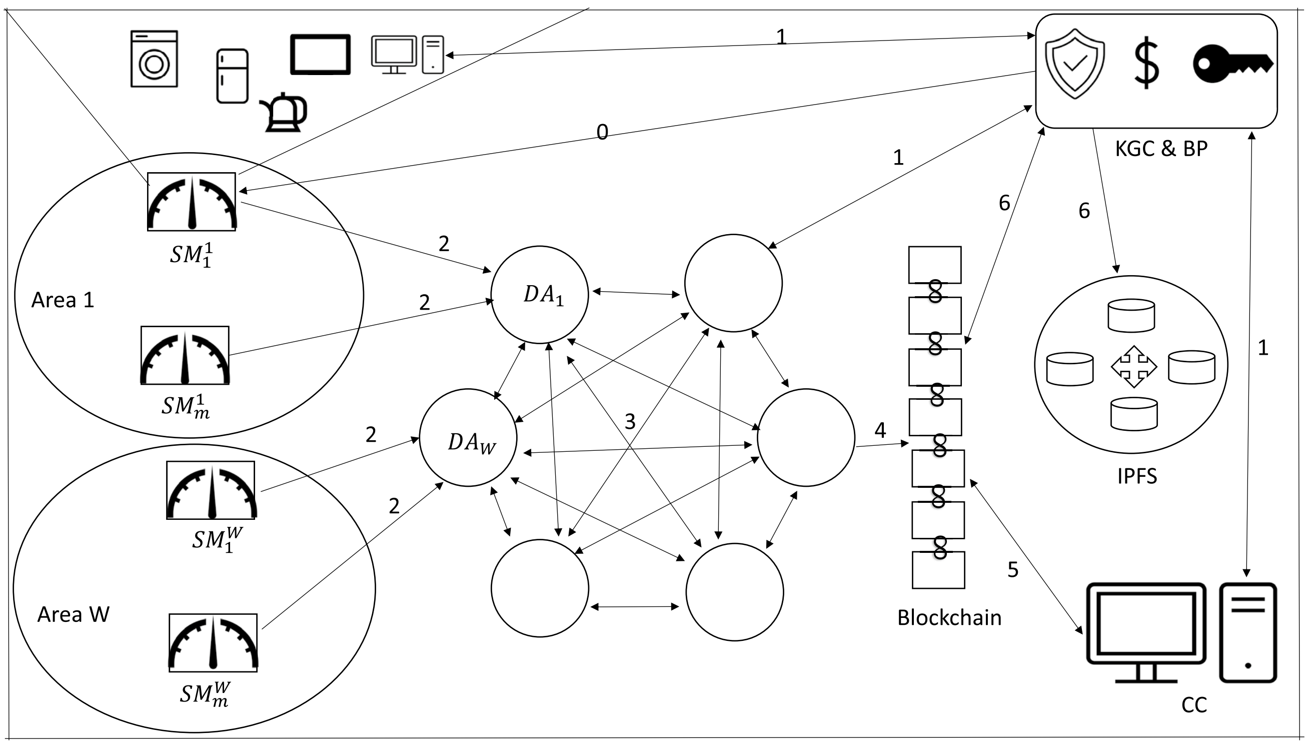

Fig. 2 System Architecture

in the blockchain. Lastly, each *U* retrieves *CID* in the blockchain and views the bill in the *IPFS*.

4.1 Threat model

In the proposed system, smart meters are trusted entities. However, the house owner can damage the smart meter by not sending any period's electricity consumption data to decrease its monthly bill. Data aggregators are malicious entities that change users' data. Moreover, they try to map anonymized identities of users to real identities to profile users. Control center is semi-honest entity that follows the protocol specification but tries to map users' anonymized identities to real identity. Key generation center and bill provider is trusted entity.

The proposed system provides the following privacy guarantees for users:

P1. Each user's sensitive data (current power consumption and bill) should not be disclosed to any unauthorized entity.

P2. Each smart meter's real identity should be kept hidden when they send electricity consumption data to an aggregator. Moreover, users' daily behavior patterns should not be linked by any adversary.

P3. Each smart meter reads user's consumption data periodically (for every 30 mins.) and sends it to data aggregator automatically. If the corresponding user tries to stop the smart meter from sending any reading, the data aggregators should catch this anomaly. This can result in a load balancing problem in the grid and incorrect billing to the corresponding user. Then, *KGCBP* should investigate this smart meter further to omit it from the grid or give a penalty.

P4. If a data aggregator tampers any aggregated consumption or any individual consumption reading, this should be caught by *DAs*.

P5. Before any transaction is added to the blockchain, *DA*s should agree on this transaction.

5 Proposed system

In this section we give the details of our system. Before delving into the details, we give high level idea about our system. *KGCBP* sets public/private key pair for each smart meter and assigns two different identities to them. Assigning two different identities to a *SM* is not new. This approach can be seen in some studies as that in [2, 3]. One identity is for billing purposes which is represented as *BID* (Bill IDentity), another one which is represented as *PRID* (Pseudo Random IDentity (public key)) for anonymously sending electricity consumption data to *DA*. The first identity is static identity such as *MAC* address of a *SM* for billing purpose. BID_i (SM_i's Bill IDentity) is the real identity of the smart meter for billing purposes. It is a unique number of the smart meter. Each user uses this number to identify their bill in the blockchain. This identity is public, and everyone can see it. This identity is used in a transaction when the bill is issued for each month. The second identity (*PRID*) is dynamic to provide anonymity for the *SM* when the *SM* sends electricity consumption data to the *DA*. A *PRID* value changes when a smart meter issues a new transaction. For example, $PRID_{j,i,k}$ represents the kth period identity of smart meter i in region j and $PRID_{j,i,k+1}$ represents the $(k+1)$th period identity of smart meter i in region j. Since each smart meter sends data to its aggregator periodically, the smart meter computes these values every period to anonymize itself. The anonymity comes from the definition of the pseudo random generator (Definition 1). We use *PRID*s as the smart meters' public keys in the protocol.

Each *SM* generates multiple (2) transactions/readings for each period and randomizes the reading in each transaction to eliminate the linkability of anonymous profiles of users. Using only single identity per transaction results in behavior linkage of the users [1, 2]. Before sending the real consumption data to the *DA*, the *SM* separates the real consumption reading to two values. One of the values is randomly chosen and the other one is the remaining value of the real consumption value. Then, two separate readings are re-randomized and sent to the *DA* with different pseudo random identities. Moreover, these identities (public keys) are generated by the *SM* using well-known cryptographic primitive, Pseudo Random Generator, *PRG* as short.

Without using this separation of each reading and re-randomization allow an adversary to link anonymous profiles of the users using users' behavior patterns, [1, 2]. For example, if a user's weekday habits are to use a kettle and toaster at 6:30 am, the power consumption data of this user are going to be the same on weekdays. It does not matter if the *SM* uses different anonymous identifiers for each electricity consumption reading.

Each smart meter sends its multiple energy consumption data for each period to break the determinism of the smart meter's transaction issue time. Deterministic behavior for a specific smart meter can be that it can send transactions every first minute of each period. To break determinism, *SM*s need to send these two readings as transactions at different times (e.g., any time between the 1st and 5th minutes of the next period). There should be a delay between the first and second transaction issue time. However, sending multiple energy consumptions as transactions to a data aggregator does not preserve unlinkability property since there is no randomness added to the energy consumption data. If the smart meter readings are not randomized, the aggregator aggregates any two received consumption values in the same region to analyze the behavior of the users. For example, it aggregates any two consumptions of a period for a region and makes a table for them. There are $2m(2m-1)/2$ different aggregations of a period for each region, where m is the total number of smart meters in a region. Thus, any untrusted party can link the identities (even if it is random identities) when the adversary has more and more information (energy consumption readings for each period) about the corresponding region. Then, it profiles users. For the unlinkability property, we add randomness to the data consumption values. Once DA_j receives two transactions that include the randomized power consumption data at different times, not sequentially, the same smart meter's transactions will not be linkable.

However, the randomization of the readings results in *CC* facing difficulty in computing each period's dynamic price. Moreover, it also results in unbalancing generated and consumed power of the region. We use a new randomization technique to randomize smart meters' real data. Then, we give a method that it lets *CC* de-randomize the readings easily. In the proposed system, a random value (mask) is generated by the smart meter for each period (based a smart meter's secret key). Then, it is added to the user's consumption data. *KGCBP* computes the smart meters' random masks for each period and region at the beginning of the protocol (**Filter Generation Phase**). Then, *KGCBP* puts them into the blockchain by the help of the data aggregators that use consensus.

The real data of a smart meter is randomly divided into two and re-randomized with random masks. Then, they are sent to the same aggregator as transactions. Each transaction includes, a pseudo random identity of smart meter (public key) that provides anonymity of the meter, a randomized electricity consumption data, a time-stamp and the signature of these values. Since the re-randomized values (consumption data) are randomly divided and re-randomized and each re-randomized consumption data is appended with a pseudo random identity once a transaction is formed, the data aggregator can not link the transactions of a smart meter. Moreover, two data transactions issued by a smart meter for a period are not sent sequentially to a data aggregator. There is a delay between two transactions to randomize the behavior of the smart meter.

Moreover, the proposed system uses two *Bloom* filters for each month. One of them (BF_1) provides fast authentication mechanism for *SM*s. BF_1 stores multiple identities of a *SM*'s each reading that helps *DA*s to check if the sender (*SM*) is authenticated and real. Furthermore, using the *PRG* function to generate Pseudo Random IDentities (*PRID*) of *SM*s eliminates heavy communication complexity between each *SM* and *KGCBP* for authenticating the *SM*s. The other *Bloom* Filter is for *DA*s to check if all authenticated *SM*s have sent their corresponding electricity consumption data of a period. The second filter (BF_2) allows *DA*s to detect faulty *SM*s that not sent their readings periodically to *DA*s. Software issues, hardware problems or malicious users prevent *SM* not to send readings. The *DA* then informs *KGCBP* about this situation. *KGCBP* then further investigate these *SM*s to eliminate the corrupted *SM*s from the system or update the *SM*s. In both cases, *Bloom* filters need to be updated. These two *Bloom* filters, BF_1 and BF_2, are sent to *DA*s and *CC*.

Furthermore, to eliminate single point of failure attack, the proposed system uses voting based [12] Practical Byzantine Fault Tolerance, *vPBFT* as short, consensus for block verification before the block is added to the blockchain. This consensus is run couple times: in block generation, in dynamic price generation and in bill report generation. Our voting based *vPBFT* is different than [12] since our scheme do not use key aggregation and encryption schemes.

In addition, tampered *SM*'s identity is being found by *KGCBP* once the block is added to the blockchain. In other words, our scheme still continue working even some faulty smart meters are presented. The corresponding *DA* informs *KGCBP* during aggregating transaction phase. *DA* sends the corresponding message $g^{x_{DA}}, AggFault, rng, area, ts$ with signature of it to *KGCBP*, $Sign_{x_{DA}}(AggFault, rng, area, ts)$, that in period *rng* and region *area*, there is an aggregation fault *AggFault*. In other words, there are *SM*s that have not sent their readings in time period *rng* and region *area*. Our system consists of 8 phases: Setup, Key Generation, Filter Generation, Data Aggregation, Consensus, Dynamic Pricing, Dynamic Billing, and Finding Faulty Smart Meter.

Setup phase In this phase, *KGCBP* sets the system parameters such as signature scheme (*Schnorr*) (G is a subgroup of Z_p^*, g is a generator of G), $2r$ hash functions for two *Bloom* filters, $HB_\ell : G \rightarrow [1, m]$, where $\ell \in \{1, \dots, r\}$, $HB'_\ell : G \rightarrow [1, m]$, where $\ell \in \{r+1, \dots, 2r\}$ and hash functions $H_1 : \{0,1\}^* \rightarrow \{0,1\}^\lambda$, $H_2 : \{0,1\}^\kappa \rightarrow Z_q^*$, a pseudo random generator (*PRG*) $G' : \{0,1\}^\kappa \rightarrow \{0,1\}^{2\kappa}$ ($f : \{0,1\}^\kappa \rightarrow \{0,1\}^\kappa$), $H_3 : \{0,1\}^\kappa \rightarrow [\phi_1, \phi_2]$ and any *CPA* secure asymmetric encryption scheme, where $[\phi_1, \phi_2]$ is a range that contains a set of integers and its endpoints are ϕ_1 and ϕ_2. H_3 maps a κ-bit value to a value in the range $[\phi_1, \phi_2]$.

Key generation phase In this phase, *KGCBP* assigns two different identities to each smart meter. These identities are for billing and anonymity purposes for SM_i. *KGCBP* assigns $BID_i \in \{0,1\}^\kappa$ for billing (public) and $s_i, s'_i, r_i, r'_i \in \{0,1\}^\kappa$ (secret) for anonymity of SM_i. *KGCBP* also assigns public/secret key pair to each *SM*. For SM_i, it assigns Pk_{SM_i} (public key) and Sk_{SM_i} (secret key). These assignments are done when the *SM* is bought by *KGCBP*. Moreover, each entity (*DA*, *U*, *CC*, and *KGCBP*) chooses a signature public and private key pair for interacting with the blockchain. This key pair for DA_i is $(SigPk_{DA_i}, SigSk_{DA_i}) = (g^{x_{DA_i}}, x_{DA_i})$, for *CC* is $(SigPk_{CC}, SigSk_{CC}) = (g^{x_{CC}}, x_{CC})$ and for *KGCBP* is $(SigPk_{KGCBP}, SigSk_{KGCBP}) = (g^{x_{KGCBP}}, x_{KGCBP})$. Moreover, public/verification key of an entity is used as identity of the entity registered by *KGCBP*.

Moreover, each entity in the system (*U*, *DA*, *CC*, *KGCBP*) chooses an asymmetric encryption key pair (*Pk*, *Sk*). Pk_{DA_j}, Sk_{DA_j} is public and secret key pair of DA_j. Similarly, each user i (U_i), *CC* and *KGCBP* chooses their public/private key pair, where $i \in \{1, \dots, N\}$ and $j \in \{1, \dots, W\}$. Each user knows their corresponding *SM*'s *BID*.

Filter generation phase *KGCBP* computes two *Bloom* filters for each month as follows (Assuming that each *SM* collects data from their home appliances for every 30 min. Thus, each *SM* does 1440 readings monthly.):

(a) For each smart meter i (SM_i) at area j, computes $t_{j,i,1} = f_{s_i}(0 \dots 000000000001)$, $t_{j,i,1} = f_{s_i}(0 \dots 000000000001)$, $z_{j,i,1} = f_{r_i}(0 \dots 000000000001)$, $z'_{j,i,1} = f_{r'_i}(0 \dots 000000000001)$ as the first readings of the first day ($t_{j,i,49} = f_{s_i}(0 \dots 000000110001)$, $t'_{j,i,49} = f_{s'_i}(0 \dots 000000110001)$, $z_{j,i,49} = f_{r_i}(0 \dots 000000110001)$, $z'_{j,i,49} = f_{r'_i}(0 \dots 000000110001)$ as the first readings of the second day). It continues to compute until 48th readings of the 30th day which is $t_{j,i,1440} = f_{s_i}(0 \dots 010110100000)$, $t'_{j,i,1440} = f_{s'_i}(0 \dots 010110100000)$, $z_{j,i,1440} = f_{r_i}(0 \dots 010110100000)$, $z'_{j,i,1440} = f_{r'_i}(0 \dots 010110100000)$.

(b) Computes $(PRID_{j,i,1} = g^{H_2(t_{j,i,1})}, PRID'_{i,1} = g^{H_2(t'_{i,1})}), \dots, (PRID_{j,i,1440} = g^{H_2(t_{j,i,1440})}, PRID'_{j,i,1440} = g^{H_2(t'_{j,i,1440})})$.

(c) Takes every $PRID_{j,i,k}$, where $k \in \{1, \dots, 1440\}$ and applies r hash functions, HB_ℓ, where $\ell \in \{1, \dots, r\}$. Then, it fills the corresponding cells with 1 s in the BF_1.

(d) Takes every $PRID'_{j,i,k}$, where $k \in \{1, \dots, 1440\}$ and applies r hash functions, HB_ℓ, where $\ell \in \{1, \dots, r\}$. Then, it fills the corresponding cells with 1 s in BF_1.

(e) Computes $T_{j,1} = \prod_{i=1}^{m}(PRID_{j,i,1}) \prod_{i=1}^{m}(PRID'_{j,i,1})$, where $T_{j,1}$ is the multiplication of at most m smart meters' *PRID*s that belongs to the area j (first index) and the same time period, the first range (second index). It then applies the result to other r hash functions HB'_ℓ, where $\ell \in \{r+1, \dots, 2r\}$. Then, it fills the corresponding cells with 1 s in BF_2.

(f) Repeats *step* − *e* 1440 times since there are 1440 different reading messages.

(g) Sends BF_1 and BF_2 to all DAs, and CC.

(h) For each region j, where $1 \leq j \leq W$, computes each smart meters' random masks $z_{j,i,k}$ and $z'_{j,i,k}$ using $PRG\ G'$ and r_i, r'_i. It also computes $Z_{j,k} = \sum_{i=1}^{m}(H_3(z_{j,i,k}) + H_3(z'_{j,i,k}))$, where the first index describes the identity of the region, and the last index describes the identity of the time period, m is the number of smart meters in a region (at most), and W is the number of regions in the system. Moreover, $Z_{j,k}$ is the result of sum of the smart meters' random masks of region j and period k, where $1 \leq j \leq W, 1 \leq k \leq 1440$. It then sends $Z_{j,k}$ values to a leader node (a DA) to add them to the blockchain after a consensus algorithm. $KGCBP$ issues a transaction $(g^{x_{KGCBP}}, Z_{j,k}, ts, Sign_{x_{KGCBP}}(H_1(Z_{j,k}, ts)))$. The consensus process is explained in *consensus phase*.

A note that $Z_{j,k}$ values are computed by $KGCBP$ in advance at the beginning of the protocol.

Data aggregation phase Each smart meter i, SM_i, in area j, reads the household appliances' electricity consumption and computes the followings (assuming that the consumption value is $c_{j,i,1}$):

(a) Chooses a random reading $c'_{j,i,1}$ from $[\phi_1, \phi_2]$. Moreover, it computes $c''_{j,i,1} = c_{j,i,1} - c'_{j,i,1}$.

(b) Computes $SPRID_{j,i,1} = H_2(t_{j,i,1})$, $SPRID'_{j,i,1} = H_2(t'_{j,i,1})$ as its signature secret keys.

(c) [re-randomization step] Computes four leaf node values for the first readings of the first day $t_{j,i,1}, t'_{j,i,1}, z_{j,i,1}, z'_{j,i,1} \in \{0,1\}^{\kappa}$ applying PRG on inputs s_i, s'_i, r_i and r'_i as that in Fig. 1, and computes $H_3(z_{j,i,1})$ and $H_3(z'_{j,i,1})$.

(d) Sends $Tx_{j,i,1} = (PRID_{j,i,1}, rc'_{j,i,1}, ts_{j,i,1}, Sign_{SPRID_{j,i,1}}(H_1(rc'_{j,i,1}, ts_{j,i,1})))$ and $Tx'_{j,i,1} = (PRID'_{j,i,1}, rc''_{j,i,1}, ts'_{j,i,1}, Sign_{SPRID'_{j,i,1}}(H_1(rc''_{j,i,1}, ts'_{j,i,1})))$ to DA_j, where $Tx_{j,i,1}$ is the first transaction of smart meter i in area j, where $rc'_{j,i,1} = c'_{j,i,1} + H_3(z_{j,i,1})$ and $rc''_{j,i,1} = c''_{j,i,1} + H_3(z'_{j,i,1})$.

To satisfy the unlinkability property, we add randomness to the data consumption values using other two secret seeds (r_i, r'_i) at $step - c$. A note that there is a random time delay between two transactions are sent by each smart meter at $step - d$ to DA_j.

A note that the smart meter's real energy consumption data for the first period is $c_{j,i,1}$. However, the smart meter sends two different consumption data: $rc'_{j,i,1}$ and $rc''_{j,i,1}$. The sum of these values equals $rc'_{j,i,1} + rc''_{j,i,1} = c_{j,i,1} + H_3(z_{j,i,1}) + H_3(z'_{j,i,1})$. The value of $H_3(z_{j,i,1}) + H_3(z'_{j,i,1})$ should be eliminated when the CC calculates the first period's energy price. Moreover, CC needs to cancel out the value of $H_3(z_{j,i,1}) + H_3(z'_{j,i,1})$ to balanced the load.

Then, DA_j checks if the sender is authenticated as follows:

1. It collects the timestamps $(ts_{j,i,k}, ts'_{j,i,k})$ of transactions issued between $[x, y]$, where $y > x$.
2. It applies $PRID_{j,i,1}$ and $PRID'_{j,i,1}$ on r hash functions, HB_{ℓ}, where $\ell \in \{1, \dots, r\}$ and checks if all r locations have 1s in BF_1. If so, DA_j checks if the signature is valid.
3. If all checks are valid from $step - 1$ and $step - 2$. Then, DA_j groups the same day and the same time range reading messages sent from the smart meters in the same area. It retrieves pseudo random identities, $PRID$, from the transactions, and computes $T_{j,1} = \prod_{i=1}^{m}(PRID_{j,i,1}) \prod_{i=1}^{m}(PRID'_{j,i,1})$, where $T_{j,1}$ is the multiplication of at most m smart meters' $PRID$s.
4. It applies $T_{j,1}$ to r hash functions, HB'_{ℓ}, where $\ell \in \{r+1, \dots, 2r\}$, checks if all r locations have 1s in BF_2. If so, DA_j starts the process of $PBFT$ consensus phase. If all locations are not 1 s in BF_2, DA_j still starts $PBFT$ consensus but it also informs $KGCBP$ that one or some of the SMs are faulty. $KGCBP$ then further investigate (finding the tampered SMs) once the block is added after the consensus phase. To inform $KGCBP$, DA_j sends the corresponding message $(g^{x_{DA_j}}, AggFault, rng, area, ts)$ with the signature of it to $KGCBP$, $Sign_{x_{DA_j}}(H_1(AggFault, rng, area, ts))$, that says in period rng and region $area$, there is an aggregation fault $AggFault$. In other words, there are SMs that have not sent their readings in period rng and region $area$.
5. Generates $(g^{x_{DA_j}}, (Tx_{j,i,1}, Tx'_{j,i,1}, \dots, Tx_{j,n,1}, Tx'_{j,n,1}), ts, Sign_{x_{DA_j}}(H_1((Tx_{j,i,1}, Tx'_{j,i,1}, \dots, Tx_{j,n,1}, Tx'_{j,n,1}), ts)))$ and sends it to the general transaction pool (GTP).

A note that these steps $(step - 1, 2, 3, 4, 5)$ are repeated 1440 times since there are 1440 readings needs to be done by each SM.

In $step - 1$, DA_j collects the transactions issued between x and y for a period. In $step - 2$, DA_j checks if the smart meter is authenticated by hashing pseudo random identities of the smart meter using r hash functions. Then, it checks if all r locations are set 1 s in BF_1. If not the smart meter is not authenticated. DA_j discards the transaction. In $step - 3$, DA_j checks if any smart meter in area j did not send their energy consumption transactions using *Bloom* filter BF_2. DA_j simply computes the multiplications of the received transactions' pseudo random identities. Then, in $step - 4$, it evaluates the result on r hash functions and checks if the r locations of BF_2 have 1 s. If not, it informs $KGCBP$ by sending a transaction that one or more smart meters are faulty that did not send their consumption transactions in time range $[x, y]$. Then it forms a transaction. In $step - 5$, DA_j sends valid transactions to the general transaction pool to be added into the blockchain.

Consensus phase In this phase, all DAs $(DA_1, DA_2, \dots, DA_W)$, where, $W > 3f + 1$, f is the number of faulty nodes) sets their identities as the follower nodes. They vote other nodes to choose a leader/primary/master node. Usually the leader node has strong communication capability and good hardware facilities. The leader selection algorithm is given in Table 3.

Once a leader is chosen (assuming that it is DA_j), the voting based *PBFT* consensus process starts.

Leader Node: The leader node DA_j does the followings:

- Takes $2m$ (the maximum number of transactions that a block stores) transactions from *GTP* formed by the same *DA*, forms the block (assuming that it is $Block_o$) that consists of block header and body illustrated in Table 4.
- Chooses a random number *rnd* and prepares vote request message, then it sends $(Block_o, (g^{x_{DA_j}}, VoteRequest, rnd, ts, Sign_{x_{DA_j}}(H_1(VoteRequest, rnd, ts))))$ to other DA_k nodes, where $j \neq k$.
- Sets *Count* as 0.

Once each DA_k receives the block and vote request at time ts^*, it checks

(A) if $ts^* - ts > 0$ and signature of vote request message is valid,
(B) transactions, signatures and authentication of *SM*s (using bloom filter BF_1) are valid,
(C) computes $T_{j,1}$, as that in *step* − 3 in *Data Aggregation Phase*. Then, it applies to r hash functions as that in *step* − 4 in *Data Aggregation Phase*. If all r locations do not consists of 1 s in BF_2, DA_k sends aggregation fault message $(g^{x_{DA_k}}, AggFault, rng, area, ts)$ with signature of it to *KGCBP*, $(Sign_{x_{DA_k}}(H_1(AggFault, rng, area, ts)))$,
(D) computes *Merkle Tree Root*, MTR^* and checks if $MTR == MTR^*$,
(E) computes current block hash (CBH^*) and checks if $CBH == CBH^*$.

Table 3 The Leader Selection Algorithm

```
1 $Follower \leftarrow AD_j\ (j \in \{1, 2, \ldots, W\})$, where $W > 3f + 1, fisthenumberoffaultynodes$,
2 Set the original number of votes to 0, $v = 0$,
3 Set a random timeout $Time_{out}$ and start a timer $Timer$,
4 while$Timer > Time_{out}$ do
5 Set $Candidate \leftarrow Follower$,
6 Start the new Timer $Timer_{new}$,
7 $v = v + 1$.
8 Generate a random number $rnd_{DA_j}$ and a timestamp $ts_{DA_j}$.
9 Compute $g^{DA_j}, VoteRequest, rnd_{DA_j}, ts_{DA_j}$ and $Sign_{x_DA_j}(H_1(VoteRequest, rnd_{DA_j}, ts_{DA_j}))$,
10 Send a request of voting to all other nodes by sending $g^{DA_j}, VoteRequest, rnd_{DA_j}, ts_{DA_j}, Sign_{x_DA_j}(H_1(VoteRequest, rnd_{DA_j}, ts_{DA_j}))$ and wait for the
   reply votes,
11 Assume that $DA_q$ receives message at time $ts_{DA_q}$: $g^{DA_j}, VoteRequest, rnd_{DA_j}, ts_{DA_j}$ and $Sign_{x_DA_j}(H_1(VoteRequest, rnd_{DA_j}, ts_{DA_j}))$. $DA_q$ does the
   followings:
12 if $((ts_{DA_j} < ts_{DA_q})$ and (the signature is valid)
13 computes $(g^{DA_q}, VoteReplay, rnd_{DA_j}, ts^*_{DA_q})$ and $Sign_{x_{DA_q}}(H_1(VoteReplay, rnd_{DA_j}, ts^*_{DA_q}))$, where $ts^*_{DA_q}$ is the time when the message is formed by
   $DA_q$, and sends the computed values to $DA_j$.
14 end if
15 for each replay message $(g^{DA_q}, VoteReplay, rnd^*_{DA_j}, ts^*_{DA_q})$ and $Sign_{x_{DA_q}}(H_1(VoteReplay, rnd^*_{DA_j}, ts^*_{DA_q}))$, where $ts^*_{DA_q}$ received by $DA_j$ at time $ts^*_{DA_j}$
   do
16 if $((ts^*_{DA_q} < ts^*_{DA_j})$ and $(rnd^*_{DA_j} = rnd_{DA_j})$ and (the signature is valid)),
17 set $v = v + 1$.
18 end if
19 end for
20 if$v > \frac{W}{2} + 1$ then
21 $Leader \leftarrow Candidate$
22 else
23 $Follower \leftarrow Candidate$
24 Repeat steps $6 - 10$ for a new election.
25 end if
26 end while
```

Table 4 Block Structure

Block Header
Block Number (BN)
Previous Block Hash (PBH)
Merkle Tree Root (MTR)
Timestamp (TS)
Block Body
Region Identity and Time Period (RITP)
Transactions (TR)
Current Block Hash (CBH)
Signature of CBH (SCBH)

If $steps - A, B, D, E$ are done successfully, DA_k sends block verification status message $(g^{x_{DA_k}}, BlockVerStatus, rnd^*, ts^{**}, Sign_{DA_k}(H_1(BlockVerStatus, rnd^*, ts^{**})))$ to the leader.

The leader node, once it receives block verification status message from each DA_k, does the followings:

- Checks if $rnd == rnd^*$ and signature of $(BlockVerStatus, rnd^*, ts^{**})$ is valid.
- Checks if $BlockVerStatus == valid$.

If all these checks are valid, then the leader node update $Count$ as 1. If there are more than $2f + 1$ valid messages, then the leader node adds the formed block to blockchain, where $W > 3f + 1$ and f is the number of faulty nodes (DA).

Once the leader gets at least $\frac{2}{3}$ of the total votes, the leader forms the block as shown in Table 4. The block gets a unique identification number (BN). To preserve immutability of the blocks, the previous block's hash (PBH) value is added to the block. For all the transactions, a *Merkle* tree is built and the root of the tree (MTR) is stored in the block. The block also has a timestamp (TS) when the block is created. These values are stored in the block header of the created block. The transactions' time period when they are received and the transactions' region where they belong to ($RITP$), and all the transactions (TS) are stored in the body of the block. A hash value (CBH) of all the information is also computed to store in the next block to preserve immutability. At last, the signature of the hash ($SCBH$) also stored in the block body.

Consensus should be implemented in the system before the transactions are added to the blockchain. The nodes (Data Aggregator (DA)) should verify each transaction based on authenticity of the sender and its signature. Then, all transactions of a region with a period are added to the blockchain as a block. A consensus consists of two communication phases: the first one is for the selection of a leader, and the second one is for the verification of a block. For the selection of a leader, the nodes elect a leader that forms the block and adds it to the blockchain. In the election, a candidate node sends its message to all other nodes and waits for their votes. If it gets more than half of the votes, the candidate node becomes the leader node. The communication complexity of the leader node in this phase $O(W)$. The detail of the leader selection algorithm is given in Table 3. In the verification of a block, the leader node sends the formed block to others and waits their responses. In this phase also takes $O(W)$.

Dynamic pricing phase In this phase, CC gets all the randomized consumption data of users for each period and retrieves random masks $Z_{j,k}$ from the blockchain. For each region $1 \le j \le W$ and period $1 \le k \le 1440$, it computes $\sum_{i=1}^{m} rc'_{j,i,k} + rc''_{j,i,k} - Z_{j,k}$. Then, it computes electricity price for each time period and region. For a month, there are 1440 different time ranges. Thus, there are 1440 different prices, $p_1, p_2, \dots, p_{1440}$. These prices are formed as transactions by CC: $(g^{x_{CC}}, p_i, rng_i, ts_i, Sign_{x_{CC}}(H_1(p_i, rng_i, ts_i)))$, where $1 \le i \le 1440$, where rng_i is ith range in a month.

Then, CC sends these transactions to a new leader. A new consensus process has been done by the same way as that in *Consensus Phase* (except $step - C$ and not using BF_1, BF_2). At the end of the process, the leader adds these transactions to the blockchain.

Dynamic billing phase In this phase, $KGCBP$ computes each user's dynamic bill. For user i, $KGCBP$

(a) Computes pseudo random identities of SM_i applying s_i, s'_i to $PRG\ G'$.
(b) Retrieves $rc'_{j,i,k} = c'_{j,i,k} + H_3(z_{j,i,k}), rc''_{j,i,k} = c''_{j,i,k} + H_3(z'_{j,i,k})$ from the blockchain, where $1 \le k \le 1440$, and de-randomizes them. To de-randomize, $KGCBP$ uses $PRG\ G'$ function and secret seeds r_i, r'_i to compute $z_{j,i,k}$ and $z'_{j,i,k}$ values. Then, it computes $c_{j,i,k} = rc'_{j,i,k} + rc''_{j,i,k} - H_3(z_{j,i,k}) - H_3(z'_{j,i,k})$ for each time period k.
(c) Computes Total Electricity Consumption of U_i, $TEC_{U_i} = \sum_{j=1}^{1440} c_{j,i,k} p_k$, where $c_{j,i,k} = c'_{j,i,k} + c''_{j,i,k}$.
(d) Generates a detailed report ($BillReport$) of bill for the user. The report consists of monthly billing history graph that charts their energy usages. Electricity price and user's electricity consumption for each time period are also included. Moreover, TEC_{U_i} is also included.
(e) Sends $Enc_{Pk_{U_i}}(BillReport_{U_i})$ (U_i's encrypted bill report) to $IPFS$. $IPFS$ returns its content identifier CID_{U_i} to $KGCBP$.
(f) For each U_i, forms a transaction $(g^{x_{KGCBP}}, BID_{SM_i}, CID_{U_i}, Hash, ts, Sign_{x_{KGCBP}}(H_1(BID_{SM_i}, CID_{U_i}, Hash, ts)$, where $Hash = H_1(Enc_{Pk_{U_i}}(BillReport_{U_i}))$.
(g) Sends these transactions to a new leader.

(h) The leader executes *Consensus Phase* above to add bills to the blockchain. Each user U_i, gets in the blockchain using their credentials. Then it does the followings:

- Search their BID_{SM_i} in the blockchain.
- Retrieves CID_{U_i} in the transaction.
- Retrieves encrypted bill from *IPFS*.
- Computes $Hash^* = H_1(Enc_{Pk_{U_i}}(BillReport_{U_i}))$ and checks if $Hash == Hash^*$.
- If so, decrypts it and view their bill reports.

Finding faulty smart meters phase *KGCBP* collects aggregation fault messages were sent from *DA*s in *Consensus Phase* at $step - C$. It does the followings for each message came from DA_k:

(a) Checks integrity of the message.
(b) Checks time period of *rng* and *area*.
(c) Checks if $AggFault == 1$.
(d) If more than $2f + 1$ nodes' *rng* and *area* values are the same, $AggFault = 1$ values are the same, and signatures are verified, checks the blockchain that which *SM*s sent transactions and which did not send for that region *area* examining that range (*rng*).
(e) Needs to check $2m$ *PRID*s to find the ones that did not send the transactions.
(f) Finds the tempered *SM*s to bill those *SM*s with a penalty or eliminates those *SM*s from the grid.
(g) Re-calculates the sum of the smart meters' random masks in a region where one or more smart meters are faulty. If the faulty smart meter is SM_i of region j at period k, *KGCBP* re-computes $Z'_{j,k} = Z_{j,k} - H_3(z_{j,i,k}) - H_3(z'_{j,i,k})$ and sends the new value ($Z'_{j,k}$) to a *DA*. The *DA* then initiates a consensus to add new $Z'_{j,k}$ value to the blockchain, where k is the period that at least one smart meter in a region j is faulty.

A note that $step - g$ is needed when at least one smart meter is faulty in a region and a period. The random masks should be updated by *KGCBP* in that region and period to allow *CC* to compute correct price of that period. If the random masks are not re-calculated and *CC* uses $Z_{j,k}$ value instead of $Z'_{j,k}$ value (updated) in **Dynamic Pricing Phase**, *CC* will calculate a wrong price for the period. This is because $Z_{j,k}$ still has a random mask of the faulty smart meter SM_i. This random value of the faulty smart meter needs to be cancelled out by *KGCBP*.

A note that the genesis block has the same form as illustrated in Table 4. However, some parts of the block are set to empty strings. Since it is the first block, the block number is set to 0 (or 1). There is no parent block of it. Thus, the previous block hash is set to an empty string. There is no transaction list in the genesis block. Thus, the transaction data field is set to an empty string. Since there is no transaction in the genesis block, *Merkle* root field is set to an empty string. In addition, region identity and period fields are set to empty strings. The genesis block should be agreed upon all other nodes. The creator of the genesis block can be the one that has the lowest (or highest) address number.

Bloom filter data structure and pseudo random generator (*PRG*) provide fast authentication for the smart meters. Using *PRG* in our system decreases the communication information between *KGCBP* and each smart meter from $O(n)$ to $O(1)$, where n is the number of identities (public keys) for a smart meter in a month. This information is needed for a smart meter to prove its identity once it issues a power consumption transaction. The receiver must check if the transaction comes from a legal smart meter. In our transaction, the smart meter is given a single secret seed, and it is used in a *PRG* function to generate n pseudo-random identities. With the help of the *Bloom* filter, the search time complexity of a given identity in a transaction for a data aggregator (*DA*) is decreased from $O(mn)$ to $O(1)$, where m is the number of smart meters in a region and n is the number of identities of a smart meter. If the *Bloom* filter is not used in the system, each smart meter uses a different identity for its anonymity once it issues a transaction, each *DA* is given $O(mn)$ information by *KGCBP*. Then, each *DA* having $O(mn)$ identities to verify if the issued transaction comes from a valid smart meter. Using the *Bloom* filter, it takes only $O(1)$ time for a *DA* to verify whether a given identity in a transaction is valid.

6 Security analysis

In this section, we show that the proposed system satisfies the requirements in Sect. 4.1.

Proving security In this section, we show that the proposed system provides transaction data privacy and transaction unlinkability (anonymity) of the smart meters (users).

There are two kinds of sensitive user data that should be protected from dishonest entities in the system. One of them is the real-time energy consumption data sent from the user's smart meter to a data aggregator. The user's real time energy consumption data, $c_{i,j}$, where smart meter i's jth period data is in the transactions $Tx_{i,j}$ and $Tx'_{i,j}$. The second sensitive data is the user's monthly data (bill). Since *KGCBP* uses a *CPA* secure encryption algorithm, the second data is already protected from any unauthorized entity. Each user U_i gets in the blockchain network to retrieves their $BillReport_{U_i}$. This value is encrypted with the user's public key. Thus, the other entities is not able to read/view the user's report in the cleartext. We show that the user's first data (real-time electricity consumption data) is protected.

We prove security using a hybrid experiment. A smart meter's transaction is the form of $Tx = [\gamma_1, \gamma_2, \gamma_3, \gamma_4]$, where γ_1 is the identity of the smart meter, γ_2 is the real-time electricity consumption data, γ_3 is the time stamp, γ_4 is the smart meter's signature on these values.

Let $Tx = [\gamma_1, \gamma_2, \gamma_3, \gamma_4]$ be the challenge transaction that given to the adversary during the real attack. R is a random value in $[\phi_1, \phi_2]$ and R' is a random element of G. We define the following hybrid games which differ on what challenge transaction is given to the adversary:

Δ_0: The challenge transaction is $Tx^0 = [\gamma_1, \gamma_2, \gamma_3, \gamma_4]$.

Δ_1: The challenge transaction is $Tx^1 = [\gamma_1, R, \gamma_3, \gamma_4]$.

Δ_2: The challenge transaction is $Tx^2 = [R', R, \gamma_3, \gamma_4]$.

A note that the challenge transaction in Δ_2 leaks no information about the identity since it consists of four random elements, whereas in Δ_0 the challenge is the real game that is well formed. We show that the transitions from Δ_0 to Δ_1 to Δ_1 to Δ_2 are all computationally indistinguishable.

Theorem 1 (Transaction Data Privacy). *If f is a pseudo random function from a length doubling pseudo random generator G', there is no adversary distinguishes between games Δ_0 and Δ_1 with advantage greater than ϵ.*

Before proving the theorem, we define a chosen plaintext attack (CPA) security of the system. The security is defined using the following game between an adversary $\mathcal{A}$ that attacks our system and a PRF attacker $\mathcal{B}$ and a PRG attacker. We prove the theorem by finding a contradiction. We first assume that $\mathcal{A}$ breaks our system with non-negligible probability ϵ. Then, there is an adversary $\mathcal{B}$ that uses $\mathcal{A}$ is a sub-routine to distinguish a PRF from a uniformly random function. Then, there is another adversary $\mathcal{C}$ that uses $\mathcal{B}$ as a sub-routine to distinguish PRG outputs from uniformly random strings. The second part follows from the hybrid argument in [19]. The game between these parties as follows:

Setup:

- $\mathcal{B}$ runs Setup Phase to generate system public parameters.
- Then, $\mathcal{B}$ sends the system parameters to $\mathcal{A}$.

Query Phase:

- $\mathcal{A}$ asks $\mathcal{B}$ q-adaptive energy consumption data queries of a user i for a period j.
- $\mathcal{B}$ receives the strings $(a_{i,j}, a'_{i,j}, b_{i,j}, b'_{i,j})$ from $\mathcal{C}$.
- $\mathcal{B}$ issues $Tx_{i,j}$ and $Tx'_{i,j}$ and sends them to $\mathcal{A}$.

Challenge:

- $\mathcal{A}$ outputs two energy consumption data c_{i_0,j_0} (user i_0's j_0th period), and c_{i_1,j_1} (user i_1's j_1th period) that it would like to challenged upon.
- $\mathcal{B}$ receives the strings from $\mathcal{C}$: $(a_{i_0,j_0}, a'_{i_0,j_0}, b_{i_0,j_0}, b'_{i_0,j_0})$ and $(a_{i_1,j_1}, a'_{i_1,j_1}, b_{i_1,j_1}, b'_{i_1,j_1})$.
- $\mathcal{B}$ flips a coin $b \in \{0, 1\}$ and forms Tx_b and Tx'_b and sends them to $\mathcal{A}$.

Guess: $\mathcal{A}$ outputs its guess $b' \in \{0, 1\}$ and wins if $b = b'$. $|\Pr[Adv_{\mathcal{A},\kappa} = 1] - \Pr[Adv'_{\mathcal{A},\kappa} = 1]| \leq \epsilon$.

Let $Adv_{\mathcal{A},\kappa}$ be the adversary's winning probability when the protocol uses pseudo random function (PRF) while $Adv'_{\mathcal{A},\kappa}$ be the adversary's winning probability when the protocol uses truly random function (RF), where ϵ is a negligible function.

Retriction: The challenge energy consumption c_{i_0,j_0} and c_{i_1,j_1} should satisfy the following: if a consumption query is $c_{i,j}$ in the query phase, $j_0 \neq j$ and $j_1 \neq j$. In other words, $\mathcal{A}$'s challenge consumption data of period j_0 of user i_0 and period j_1 of user i_1 should not be queried before. Otherwise, the adversary wins the game with probability 1. The attack happens when the adversary asks $c_{i,j} = x$ in the query phase and $c_{i,j} = y$ in the challenge phase. When $\mathcal{B}$ returns $rc^q_{i,j}, rc^c_{i,j}, rc^{q'}_{i,j}, rc^{c'}_{i,j}$ in the transactions, where q indicates that it is obtained from the query phase and c indicates that it is obtained from challenge phase. The adversary computes $A = rc^q_{i,j} + rc^{q'}_{i,j}$ from the query phase and $B = rc^c_{i,j} + rc^{c'}_{i,j}$ from the challenge phase. If $|A - B| = |x - y|$, then adversary guesses the given challenge transaction correctly. The subtraction eliminates random masks.

Proof **Setup:**

- $\mathcal{B}$ runs *Setup Phase* to generate system public parameters: $G', H_1, H_2, H_3, param_{Signature}, param_{encryption}$, hash functions for *Bloom* filter HB_α, HB_β, where $\alpha = [1, \ldots, r]$ and $\beta = [r+1, \ldots, 2r]$. Then it sends them to $\mathcal{A}$.

Query Phase:

- $\mathcal{A}$ issues q queries adaptively an energy consumption data of a user i for a period j, $c_{i,j}$.

- To form the transactions that consist of pseudo random identities, and $c_{i,j}$, $\mathcal{B}$ asks $\mathcal{C}$ to have period jth values.
- $\mathcal{C}$ gives $a_{i,j}, a'_{i,j}, b_{i,j}, b'_{i,j}$ to $\mathcal{B}$.
- $\mathcal{B}$ forms the transactions: $Tx_{i,j}, Tx'_{i,j}$ using $(a_{i,j}, a'_{i,j}, b_{i,j}, b'_{i,j})$. $\mathcal{B}$ computes $H_2(a_{i,j}) = SPRID_{i,j}$, $H_2(a'_{i,j}) = SPRID'_{i,j}$, $b_{i,j} = z_{i,j}$, $b'_{i,j} = z'_{i,j}$. Then, $\mathcal{B}$ forms the transactions: $Tx_{i,j} = PRID_{i,j}, rc'_{i,j}, ts_{i,j}, Sign_{SPRID_{i,j}}(H_1(rc'_{i,j}, ts_{i,j}))$ and $Tx'_{i,j} = PRID'_{i,j}, rc''_{i,j}, ts'_{i,j}, Sign_{SPRID'_{i,j}}(H_1(rc''_{i,j}, ts'_{i,j}))$, where $rc'_{i,j} = c'_{i,j} + H_3(b_{i,j})$, $rc''_{i,j} = c''_{i,j} + H_3(b'_{i,j})$, $c_{i,j} = c'_{i,j} + c''_{i,j}$, $c'_{i,j}$ is randomly chosen in $[\phi_1, \phi_2]$.

Challenge:

- $\mathcal{A}$ outputs two energy consumption data, c_{i_0,j_0} and c_{i_0,j_0}. $\mathcal{B}$ asks $\mathcal{C}$ to get masks of user i_0's j_0th period and user i_1's j_1th period. $\mathcal{C}$ returns with $a_{i_0,j_0}, a'_{i_0,j_0}, b_{i_0,j_0}, b'_{i_0,j_0}$ and $a_{i_1,j_1}, a'_{i_1,j_1}, b_{i_1,j_1}, b'_{i_1,j_1}$. $\mathcal{B}$ flips a coin $b \in \{0,1\}$, it issues Tx_b and sends it to $\mathcal{A}$. If $b = 0$, $\mathcal{B}$ forms the transaction, $Tx_0 = PRID_{i_0,j_0}, rc'_{i_0,j_0}, ts_{i_0,j_0}, Sign_{SRID_{i_0,j_0}}(H_1(rc'_{i_0,j_0}, ts_{i_0,j_0}))$, and $Tx'_0 = PRID'_{i_0,j_0}, rc''_{i_0,j_0}, ts'_{i_0,j_0}, Sign_{SRID'_{i_0,j_0}}(H_1(rc''_{i_0,j_0}, ts'_{i_0,j_0}))$. Otherwise, it sends $Tx_1 = PRID_{i_1,j_1}, rc'_{i_1,j_1}, ts_{i_1,j_1}, Sign_{SRID_{i_1,j_1}}(H_1(rc'_{i_1,j_1}, ts_{i_1,j_1}))$, and $Tx'_1 = PRID'_{i_1,j_1}, rc''_{i_1,j_1}, ts'_{i_1,j_1}, Sign_{SRID'_{i_1,j_1}}(H_1(rc''_{i_1,j_1}, ts'_{i_1,j_1}))$.

Guess: $\mathcal{A}$ outputs its guess b'.

If the masks are the outputs of the PRFs, $\mathcal{B}$ plays game Δ_0 with $\mathcal{A}$, $Adv_{\mathcal{A},\kappa} = \Pr[\mathcal{B}^{PRF(\cdot)}(1^\kappa) = 0]$, where $PRF \leftarrow \{0,1\}^\kappa$. If the masks are the outputs of uniformly random functions (RF), $\mathcal{B}$ plays game Δ_1 with $\mathcal{A}$, $RF \leftarrow F$, $Adv'_{\mathcal{A},\kappa} = \Pr[\mathcal{B}^{RF(\cdot)}(1^\kappa) = 0]$, where $RF : \{0,1\}^\kappa \rightarrow \{0,1\}^\kappa$.

Thus, we can write $|\Delta_0 - \Delta_1| = |Adv_{\mathcal{A},\kappa} - Adv'_{\mathcal{A},\kappa}| = |\Pr[\mathcal{B}^{f_{PRF}(\cdot)}(1^\kappa) = 0] - \Pr[\mathcal{B}^{RF(\cdot)}(1^\kappa) = 0]| \geq \epsilon$. It means that if $\mathcal{A}$ breaks our scheme with non-negligible probability ϵ, $\mathcal{B}$ distinguishes PRF functions from random functions. we also know the result from [19] that the relation between winning probabilities between $Adv_{\mathcal{C},\kappa}$ and $Adv_{\mathcal{B},\kappa}$ is $Adv_{\mathcal{C},\kappa} = \frac{\epsilon}{\kappa q} Adv_{\mathcal{B},\kappa}$, that PRG-attacker $\mathcal{C}$ distinguishes random strings from pseudo random strings generated by PRG G' is also non-negligible. Thus, it results in a contradiction that violates the security of G'.

The probability of $Adv'_{\mathcal{A},\kappa}$ needs to be calculated when $\mathcal{B}$ uses random strings to answer queries from $\mathcal{A}$. There are q queries. If $\mathcal{B}$ uses the same strings to answer $\mathcal{A}$'s query, it is possible that $\mathcal{A}$ figures out whether c_{i_0,j_0} or c_{i_1,j_1}. The probability of picking the same string is $\frac{q}{2^\kappa}$. Thus, $Adv'_{\mathcal{A},\kappa} = \frac{1}{2} + \frac{q}{2^\kappa}$. When we combine this result with $|\Pr[Adv_{\mathcal{A},\kappa} = 1] - \Pr[Adv'_{\mathcal{A},\kappa} = 1]| \leq \epsilon$, we have $Adv_{\mathcal{A},\kappa} = \frac{1}{2} + \frac{q}{2^\kappa} + \epsilon = \frac{1}{2} + \epsilon'$, where $\epsilon' = \frac{q}{\kappa} + \epsilon$, ϵ' is a negligible function. □

Thus, the first privacy guarantee $P1$ is satisfied in Sect. 4.1.

Theorem 2 (Transaction (Smart Meter identity) *Unlinkability*). *If f is a pseudo random function PRF from a length doubling pseudo random generator PRG G', there is no adversary distinguishes between games Δ_1 and Δ_2 with non-negligible probability ϵ.*

We prove this theorem using its contrapositive. Supposing that there is an adversary $\mathcal{A}$ links a smart meter's identities in the transactions with negligible probability ϵ. Then, there exists an adversary $\mathcal{B}$ uses $\mathcal{A}$ as a sub-routine to distinguish a pseudo random function PRF f from a random function g with non-negligible probability. Then, there is another adversary $\mathcal{C}$ that uses $\mathcal{B}$ to distinguish random strings from pseudo random strings generated by PRG G' with non-negligible probability. $\mathcal{B}$ receives some identity queries from $\mathcal{A}$. Then, $\mathcal{B}$ obtains the identities from $\mathcal{C}$. The second part of the proof follows from the proof [19].

Before proving this theorem, we define a security game between PRG attacker $\mathcal{C}$, PRF attacker $\mathcal{B}$ and attacker $\mathcal{A}$ as follows:

Setup:

- $\mathcal{B}$ runs Setup Phase to generate system public parameters.
- Then, $\mathcal{B}$ sends the system parameters to $\mathcal{A}$.

Query Phase:

- $\mathcal{A}$ asks $\mathcal{B}$ q-adaptive identity queries of a user i for a period j.
- $\mathcal{B}$ receives the strings $(a_{i,j}, a'_{i,j}$ from $\mathcal{C}$.
- $\mathcal{B}$ issues $Tx_{i,j}$ and $Tx'_{i,j}$ and sends them to $\mathcal{A}$.

Challenge:

- $\mathcal{A}$ outputs two identity and period tuples id_{i_0,j_0} (user i_0's j_0th period), and id_{i_1,j_1} (user i_1's j_1th period) that it would like to challenged upon.
- $\mathcal{B}$ receives the strings from $\mathcal{C}$: $(a_{i_0,j_0}, a'_{i_0,j_0})$ and $(a_{i_1,j_1}, a'_{i_1,j_1})$.
- $\mathcal{B}$ flips a coin $b \in \{0,1\}$ and forms Tx_b and Tx'_b and sends them to $\mathcal{A}$.

Guess: $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$ and wins if $b = b'$. $|\Pr[Adv_{\mathcal{A},\kappa} = 1] - \Pr[Adv'_{\mathcal{A},\kappa} = 1]| \leq \epsilon$.

Let $Adv_{\mathcal{A},\kappa}$ be the adversary's winning probability when the protocol uses pseudo random function (PRF) while $Adv'_{\mathcal{A},\kappa}$ be the adversary's winning probability when the protocol uses truly random function (RF), where ϵ is a function of κ and negligible.

Retriction: The challenge identity and period tuples id_{i_0,j_0} and id_{i_1,j_1} should satisfy the following: if an identity query is $id_{i,j}$ in the query phase, $j_0 \neq j$ and $j_1 \neq j$. In other words, $\mathcal{A}$'s challenge identity of period j_0 of user i_0 and period j_1 of user i_1 should not be queried before. Otherwise, the adversary wins the game with probability 1.

Setup:

- $\mathcal{B}$ runs *Setup Phase* to generate system public parameters and gives them to $\mathcal{A}$.

Query Phase:

- $\mathcal{A}$ makes q adaptive transaction identity queries of a smart meter i for a time period j ($id_{i,j}, id'_{i,j}$), where $j = 1, \dots, q$.
- To form the transactions that consist of the smart meter's identities, $\mathcal{B}$ asks $\mathcal{C}$ to have them.
- $\mathcal{C}$ gives $a_{i,j}, a'_{i,j}$ to $\mathcal{B}$.
- $\mathcal{B}$ forms the transactions $Tx_{i,j}, Tx'_{i,j}$ by using $(a_{i,j}, a'_{i,j})$ values. $\mathcal{B}$ computes $H_2(a_{i,j}) = SPRID_{i,j}, H_2(a'_{i,j}) = SPRID'_{i,j}$. Then, $\mathcal{B}$ computes $Tx_{i,j} = PRID_{i,j}, rc'_{i,j}, ts_{i,j}, Sign_{SPRID_{i,j}}$ $(H_1(rc'_{i,j}, ts_{i,j}))$ and $Tx'_{i,j} = PRID'_{i,j}, rc''_{i,j}, ts'_{i,j}, Sign_{SPRID'_{i,j}}$ $(H_1(rc''_{i,j}, ts'_{i,j}))$, where $rc'_{i,j}, rc''_{i,j}$ are random values chosen from $[\phi_1, \phi_2]$.

Challenge:

- $\mathcal{A}$ outputs two identities id_{i_0,j_0} and id_{i_1,j_1}, where $j_0 \neq j$ and $j_1 \neq j$. $\mathcal{B}$ forwards these identities $id_{i_0,j_0}, id_{i_1,j_1}$ to $\mathcal{C}$. $\mathcal{C}$ returns with $a_{i_0,j_0}, a'_{i_0,j_0}$ and $a_{i_1,j_1}, a'_{i_1,j_1}$. $\mathcal{B}$ flips a coin $b \in \{0, 1\}$, it issues Tx_b and sends it to $\mathcal{A}$. If $b = 0$, $\mathcal{B}$ forms the transaction, $Tx_0 = PRID_{i_0,j_0}, rc'_{i_0,j_0}, ts_{i_0,j_0}, Sign_{SRID_{i_0,j_0}}(H_1(rc'_{i_0,j_0}, ts_{i_0,j_0}))$, and $Tx'_0 = PRID'_{i_0,j_0}, rc''_{i_0,j_0}, ts'_{i_0,j_0}$, $Sign_{SRID'_{i_0,j_0}}(H_1(rc''_{i_0,j_0}, ts'_{i_0,j_0}))$. Otherwise, it sends $Tx_1 = PRID_{i_1,j_1}, rc'_{i_1,j_1}, ts_{i_1,j_1}, Sign_{SRID_{i_1,j_1}}(H_1(rc'_{i_1,j_1}, ts_{i_1,j_1}))$, and $Tx'_1 = PRID'_{i_1,j_1}, rc''_{i_1,j_1}, ts'_{i_1,j_1}, Sign_{SRID'_{i_1,j_1}}(H_1(rc''_{i_1,j_1}, ts'_{i_1,j_1}))$, where $rc'_{i_0,j_0}, rc''_{i_0,j_0}, rc'_{i_1,j_1}, rc''_{i_1,j_1}$ are random values chosen from $[\phi_1, \phi_2]$.

Guess: $\mathcal{A}$ outputs its guess b'.

If the masks are the outputs of the *PRF*s, $\mathcal{B}$ plays game Δ_1 with $\mathcal{A}$, $Adv_{\mathcal{A},\kappa} = \Pr[\mathcal{B}^{PRF(\cdot)}(1^\kappa) = 0]$, where $PRF \leftarrow \{0, 1\}^\kappa$. If the masks are the outputs of uniformly random functions (*RF*), $\mathcal{B}$ plays game Δ_2 with $\mathcal{A}$, $RF \leftarrow F$, $Adv'_{\mathcal{A},\kappa} = \Pr[\mathcal{B}^{RF(\cdot)}(1^\kappa) = 0]$, where $RF : \{0, 1\}^\kappa \rightarrow \{0, 1\}^\kappa$.

Thus, we can write $|\Delta_0 - \Delta_1| = |Adv_{\mathcal{A},\kappa} - Adv'_{\mathcal{A},\kappa}| = |\Pr[\mathcal{B}^{f_{PRF}(\cdot)}(1^\kappa) = 0] - \Pr[\mathcal{B}^{RF(\cdot)}(1^\kappa) = 0]| \geq \epsilon$. It means that if $\mathcal{A}$ breaks our scheme with non-negligible probability ϵ, $\mathcal{B}$ distinguishes *PRF* functions from random functions. we also know the result from [19] that the relation between winning probabilities between $Adv_{\mathcal{C},\kappa}$ and $Adv_{\mathcal{B},\kappa}$ is $Adv_{\mathcal{C},\kappa} = \frac{\epsilon}{\kappa q} Adv_{\mathcal{B},\kappa}$, that *PRG*-attacker $\mathcal{C}$ distinguishes random strings from pseudo random strings strings generated by *PRG* G' is also non-negligible. Thus, it results in a contradiction that violates the security of G'.

The probability of $Adv'_{\mathcal{A},\kappa}$ needs to be calculated when $\mathcal{B}$ uses random strings to answer queries from $\mathcal{A}$. There are q queries. If $\mathcal{B}$ uses the same strings to answer $\mathcal{A}$'s query, it is possible that $\mathcal{A}$ figures out whether id_{i_0,j_0} or id_{i_1,j_1}. The probability of picking the same string is $\frac{q}{2^\kappa}$. Thus, $Adv'_{\mathcal{A},\kappa} = \frac{1}{2} + \frac{q}{2^\kappa}$. When we combine this result with $|\Pr[Adv_{\mathcal{A},\kappa} = 1] - \Pr[Adv'_{\mathcal{A},\kappa} = 1]| \leq \epsilon$, we have $Adv_{\mathcal{A},\kappa} = \frac{1}{2} + \frac{q}{2^\kappa} + \epsilon = \frac{1}{2} + \epsilon'$, where ϵ' is a negligible function.

Thus, the second privacy guarantee ($P2$) is satisfied in Sect. 4.1.

If SM_i does not send its reading for a time range to DA_j, other entities DA_j and *CC* catch this behavior and inform *KGCBP*. The proposed system uses Bloom filter BF_2 for this situation. Thus, the proposed system satisfies $P3$ in Sect. 4.1.

If a malicious *DA* changes any received reading from a *SM*, this will be caught by other *DA*s using Bloom filter BF_1. Thus, $P4$ is observed in Sect. 4.1.

Before adding transactions to the blockchain, the proposed system uses voting based *PBFT* consensus. Using consortium blockchain eliminates single point of failure problem in the system. This satisfies $P5$ in Sect. 4.1.

7 Discussions

In this section, we give some discussions about the proposed system. In the proposed system, each smart meter's keys s, s', r, r' are used for only a month. For other months, *KGCBP* needs to send different (fresh) random keys to each smart meter. The new keys can be adjusted as follows:

- onsite adjustment: *KGCBP* assigns new keys offline.
- *KGCBP* encrypts new s, s' values using the smart meter's public key and sends them to it. Then each *SM* decrypts it to have new keys.
- each *SM* and *KGCBP* use hash based key derivation function (*HKDF*) to have a common key [24] for other months.

In the proposed system, *KGCBP* needs to do offline computation (checking) to find smart meters that did not send their consumption data in a period. For each period, there are at most $2m$ transactions consisting of the users' *PRID*s. To find faulty smart meters, *KGCBP* can locally prepare a table with hashes of *PRID*s for each time range. Then, it retrieves the *SM*s' *PRID*s from the blockchain to check them to determine which smart meters did not send their readings. Once *KGCBP* figures out the faulty *SM*s, it then needs to be sure which ones have been tampered by checking the log files in the smart meters.

Consumers/users can do data tampering on their *SM*s to send less electricity consumption data than it needs to be. This results another load unbalancing and incorrect billing. To find these tempered *SM*s, *KGCBP* can do detect these *SM*s using Artificial Intelligence (AI) and Machine Learning (ML) methods [25].

For real world application, a complete randomization of power consumption $c'_{i,j}$ is important. In study [26], random slices are chosen from range $[-2^{31}, 2^{31}]$. That means, random slices are represented as 32 bits. We can set $[\phi_1, \phi_2]$ as $[-2^{31}, 2^{31}]$.

In the proposed protocol, if there is no faulty smart meter in any region and period, *KGCBP* does not need to re-compute the random masks of the smart meters of the region and period that a smart meter was faulty. However, if a smart meter is faulty for a period in a region, *KGCBP* should update the aggregated random masks of that period in the region.

Although blockchain can bring several advantages, it sacrifices the efficiency (scalability). The efficiency is an inherit problem for all blockchains. The efficiency problem comes from consensus algorithm. Efficiency problem comes from the consensus protocol. To form a block that consists of users' power consumption transactions, the block should be verified by the nodes in the network. The leader node broadcasts the block and waits its peers' responses. Once it gets the majority of responses, it adds the block into blockchain. Beside block addition phase, there is a leader selection phase.

8 Conclusion

In this paper, we proposed a blockchain based secure and efficient data aggregation and dynamic billing system. The proposed system provides fast authentication for smart meters that allows data aggregators to check if the received messages are issued from real smart meters. Moreover, the system allows entities to find out if smart meters did not send their periodic electricity consumption data. To eliminate single point of failure, the proposed system uses consortium blockchain and voting based *PBFT* consensus. Furthermore, once the users' bill reports are appended to the blockchain, they search their corresponding smart meters' identities from the blockchain, retrieves their corresponding *CID*s from the blockchain and retrieves the encrypted report from the *IPFS*. Then, they decrypt them using their secret key to view their reports. This provides data privacy and eliminates scalability issue for the users in the blockchain.

As a future work, we implement our system to evaluate its performance. Moreover, we would like to find more efficient method for figuring out tempered smart meters.

Author contributions The author is responsible for the followings in the paper: Conceptualization, Methodology, Formal analysis, Writing - original draft preparation.

Funding Open access funding provided by the Scientific and Technological Research Council of Türkiye (TÜBİTAK). No funds, grants, or other support was received.

Data availability Not applicable.

Declarations

Ethics approval Not applicable.

Consent to publish Not applicable.

Competing interest The authors declare no competing interests.

References

1. Jawurek M, Johns M, Rieck K (2011) Smart metering de-pseudonymization. In: Proceedings of the 27th Annual Computer Security Applications Conference. ACSAC '11, Association for Computing Machinery, New York, NY, USA, pp 227–236. https://doi.org/10.1145/2076732.2076764
2. Cleemput S, Mustafa MA, Marin E, Preneel B (2018) De-pseudonymization of smart metering data: Analysis and countermeasures. In: 2018 Global Internet of Things Summit (GIoTS), IEEE, pp 1–6
3. Efthymiou C, Kalogridis G (2010) Smart grid privacy via anonymization of smart metering data. In: 2010 First IEEE International Conference on Smart Grid Communications, IEEE, pp 238–243
4. Guan Z, Si G, Zhang X, Wu L, Guizani N, Du X, Ma Y (2018) Privacy-preserving and efficient aggregation based on blockchain for power grid communications in smart communities. IEEE Commun Mag 56(7):82–88
5. Fan M, Zhang X (2019) Consortium blockchain based data aggregation and regulation mechanism for smart grid. IEEE Access 7:35929–35940
6. Wu F, Li X, Xu L, Kumari S, Lin D, Rodrigues JJ (2020) An anonymous and identity-trackable data transmission scheme for smart grid under smart city notion. Ann Telecommun 75:307–317

7. Zhang S, Rong J, Wang B (2020) A privacy protection scheme of smart meter for decentralized smart home environment based on consortium blockchain. Int J Electr Power Energy Syst 121:106140
8. Wagh GS, Gupta S, Mishra S (2020) A distributed privacy preserving framework for the smart grid. In: 2020 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), IEEE, pp 1–5
9. Fan H, Liu Y, Zeng Z (2020) Decentralized privacy-preserving data aggregation scheme for smart grid based on blockchain. Sensors 20(18):5282
10. Oksuz O (2020) Privacy preserving data aggregation and dynamic billing system in smart grid using permissioned blockchain. In: CS & IT Conference Proceedings, vol. 10, CS & IT Conference Proceedings
11. Oksuz O (2020) Providing anonymous communication, privacy-preserving data aggregation and dynamic billing system in smart grid using permissioned blockchain. Int J Netw Secur Appl (IJNSA) 12
12. Bera B, Saha S, Das AK, Vasilakos AV (2020) Designing blockchain-based access control protocol in iot-enabled smart-grid system. IEEE Internet Things J 8(7):5744–5761
13. Abouyoussef M, Ismail M (2021) Blockchain-based networking strategy for privacy-preserving demand side management. In: ICC 2021-IEEE International Conference on Communications, IEEE, pp 1–6
14. Bao H, Ren B, Li B, Kong Q (2021) Bbnp: a blockchain-based novel paradigm for fair and secure smart grid communications. IEEE Internet Things J 9(15):12984–12996
15. Luo X, Xue K, Xu J, Sun Q, Zhang Y (2021) Blockchain based secure data aggregation and distributed power dispatching for microgrids. IEEE Trans Smart Grid 12(6):5268–5279
16. Zhao M, Ding Y, Tang S, Liang H, Wang H (2022) A blockchain-based framework for privacy-preserving and verifiable billing in smart grid. Peer-to-Peer Netw Appl 1–14
17. Wang H, Wang L, Wen M, Chen K, Luo Y (2022) A lightweight certificateless aggregate ring signature scheme for privacy protection in smart grids. Wireless Pers Commun 126(2):1577–1599
18. Zhang X, You L, Hu G (2022) An efficient and robust multidimensional data aggregation scheme for smart grid based on blockchain. IEEE Trans Netw Serv Manag
19. Goldreich O, Goldwasser S, Micali S (1986) How to construct random functions. J ACM (JACM) 33(4):792–807
20. Rivest RL, Shamir A, Adleman L (1978) A method for obtaining digital signatures and public-key cryptosystems. Commun ACM 21(2):120–126
21. Paillier P (1999) Public-key cryptosystems based on composite degree residuosity classes. In: Stern J (ed) International Conference on the Theory and Applications of Cryptographic Techniques. Springer, Prague, Czech Republic, pp 223–238
22. Schnorr C-P (1991) Efficient signature generation by smart cards. J Cryptol 4:161–174
23. Castro M, Liskov B (2002) Practical byzantine fault tolerance and proactive recovery. ACM Trans Comput Syst 20(4):398–461. https://doi.org/10.1145/571637.571640
24. Krawczyk H (2010) Cryptographic extraction and key derivation: The hkdf scheme. In: CRYPTO, vol. 6223, Springer, pp 631–648
25. Khan ZA, Adil M, Javaid N, Saqib MN, Shafiq M, Choi J-G (2020) Electricity theft detection using supervised learning techniques on smart meter data. Sustainability 12(19). https://doi.org/10.3390/su12198023
26. Finster S, Baumgart I (2014) SMART-ER: Peer-based Privacy for Smart Metering. In: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp 652–657

Dr. Ozgur Oksuz received his PhD degree under the supervision of Prof. Aggelos Kiayias and Prof. Bing Wang from the University of Connecticut in 2016. Then, he was a postdoctoral researcher at Washington State University for a year. His host was Prof. Adam Hahn. His research areas include theoretical and applied cryptography, computer security and privacy, and blockchain technology. He is currently an Assistant Professor at Konya Technical University.