Article

Publicly Verifiable Spatial and Temporal Aggregation Scheme Against Malicious Aggregator in Smart Grid

Lei Zhang ¹ and Jing Zhang ^{2,1,*}

¹ College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China; lei_power@hrbeu.edu.cn

² School of Information Science and Engineering, Jinan University, Jinan 250022, China

* Correspondence: ise_zhangjing@ujn.edu.cn

Received: 16 November 2018; Accepted: 14 January 2019; Published: 31 January 2019

Abstract: We propose a privacy-preserving aggregation scheme under a malicious attacks model, in which the aggregator may forge householders' billing, or a neighborhood aggregation data, or collude with compromised smart meters to reveal object householders' fine-grained data. The scheme can generate spatially total consumption in a neighborhood at a timestamp and temporally a householder's billing in a series of timestamps. The proposed encryption scheme of imposing masking keys from pseudo-random function (PRF) between pairwise nodes on partitioned data ensures the confidentiality of individual fine-grained data, and fends off the power theft of $n-2$ smart meters at most (n is the group size of smart meters in a neighborhood). Compared with the afore-mentioned methods of public key encryption in most related literatures, the simple and lightweight combination of PRF with modular addition not only is customized to the specific needs of smart grid, but also facilitates any node's verification for local aggregation or global aggregation with low cost overhead. The publicly verifiable scenarios are very important for self-sufficient, remote places, which can only afford renewable energy and can manage its own energy price according to the energy consumption circumstance in a neighborhood.

Keywords: smart metering; spatial and temporal aggregation; privacy protection; internal attack; pseudo-random function

1. Introduction

With the development of Advanced Metering Infrastructure (AMI), Smart Metering as an important research subject in Smart Grid (SG) plays an increasingly important role and is closely associated with people's daily life [1,2]. Aggregating fine-grained metering data attracts householders and power suppliers. Power suppliers can calculate, forecast, and regulate accurately power distribution/price of the next period in real time while detecting fraud reports. Based on billing details and current power price, householders can adjust its appliance consumption module to reduce the power billing at the peak time; however, accessing householder's information on metering may cause security and privacy concerns, such as daily routines, the type of applications, etc. [1,2]. For this, in SG systems, one of the challenges faced by power big data is how to design one aggregation mechanism to balance the use of power data and individual privacy protection [2].

Protecting such sensitive private data from individual privacy threats needs to limit the authority of the utility company employee [2]. Namely, Supplier Billing System (SBS, sub-suppliers) will know only the total amount of the consumption for each customer, while the Energy Management System (EMS, demand prediction division) should know only the total consumption of customers in a certain region for each time period. To achieving the goals, smart metering systems often introduce the Meter

Data Management System (MDMS), which stores the measured values of smart meters (SMs), and aggregates it before sending the aggregation to the SBS and EMS [2].

With the appearance of MDMS, another concern is upgrading, namely the malicious action of householders and regional MDMS employees. Unfortunately, a malicious householder may collude with the regional MDMS employee to report a false consumption to the SBS department; attackers may steal or forge power usage and consumption information. In addition, a regional MDMS employee may submit a fraudulent aggregation in a neighborhood. A World Bank report finds that each year over 6 billion dollars cannot post due to the energy theft and fraud report in the United States, in 2009, the FBI reported a wide and organized attempt that may have cost up to \$400 million loss annually and power supplier suffered a great monetary loss [3]. To fend off this type of attack, it is desirable that suppliers or the public should detect the fraud profile from malicious aggregators or dishonest householders [4].

Privacy-preserving metering protocols have been discussed in lots of literatures [5–24]. They mainly focused on the studies of homomorphic aggregation [6–17,20,23,24], by which, aggregators can only obtain the fine-grained aggregation data within a certain region or householders' billing in a serial period while protecting individual privacy. However, most of them can only resist against single external or semi-trustable attack [11,13,14,19], and how to fend off internal attackers (e.g. aggregators or householders) is an open problem. Internal attackers can legally collect and store power consumption information of users; therefore, they pose a higher threat than external attackers [18].

Most of existing works [5–10,18,20,21,23,24] about additively homomorphic, multiplicative homomorphic, and their combination with other cryptography endeavored to address the problem. Most of them improved Paillier encryption [6–9] by their combination with other cryptography, such as stream ciphers [5,19,23] and modular addition [7,24], to prevent power suppliers/operators from intercepting individual user data, and to detect fraudulent from dishonest users. To ensure the integrity of transmitted messages and fend off attacks such as man-in-the middle attack and denial-of-service attack against SG, signature and authentication methods are proposed in References [8,15–17].

Lu et al. [6] proposed a privacy-preserving, multi-dimensional metering aggregation scheme in a neighborhood-wide grid with piallier encryption, bilinear pairing and computational Diffie-Hellman (DH) methods. For resisting against internal attackers possessing private keys, Xiao [8] introduced a spatial and temporal aggregation and authentication scheme by randomizing Paillier encryption with Lagrange interpolation. Their protocol requires $O(n^2)$ bytes of inter-action between the individual meters as well as relatively expensive cryptography on the meters (public key encryption). Chen [9] also improved Paillier encryption and proposed a privacy-preserving aggregation scheme resisting at most t compromised servers in a control center with threshold protocol.

Dimitriou et al. [20] provided a verifiable publicly aggregation scheme against dishonest users that attempt to provide fraudulent data. Any user node in the community can prove its computation accuracy by zero-knowledge proof that the two encrypted message with different public keys corresponding to the same plaintext message. While we can prove our scheme costs lower overhead to resist fraudulent report from internal nodes.

Erkin et al. [23] adopted a stream cipher (e.g. RC4) to generate pseudo-random keys as masking keys between nodes to prevent internal nodes from possessing private keys. During the aggregation within a neighborhood, all masking random keys cancelled out and the aggregation value is revealed without compromising individual privacy based on the security properties of the Paillier encryption and stream cipher. We follow its Pseudo-Random Function and combine it with modular addition. The main difference from ours is they impose the random keys from PRF on the plaintext before encrypting it with Paillier cryptography, and send the encrypted message to all nodes. We set a security parameter k to represent the number of communicate nodes in a neighborhood and improve the encryption method by replacing the costly Paillier encryption with the simple and lightweight combination. More significantly, we supplement a publicly verifiable property to detect the fraudulent profile from malicious aggregators or dishonest user nodes.

Castelluccia et al. [19] protected individual data by imposing masking keys from RC4 on the plaintext data under the multi-level wireless sensors network model. However, the protection protocol cannot resist malicious aggregators, as the session keys are generated by the sink as the aggregator. We extend its PRF method into the peer-to-peer system model and propose a privacy-preserving scheme against maliciously internal attack.

In addition, traditional modular addition was adopted in [7,24] by partitioning individual plaintext data into n shares and exchanging them between nodes (n is the number size of users in a neighborhood). Flavio et al. [7] adopted Paillier encryption and modular addition, in which every user node partitions its meter reading into n shares and transmits the encrypted shares with different public keys to the aggregator, which aggregates the data with the same public key before sending the aggregation to the users. Finally, the aggregator collects the plaintext sums to obtain the final aggregation. The method is privacy-preserving; however, during each spatial aggregation, three message exchanges are required between every user and the aggregator. Thus, the number of homomorphic encryption per user increases linearly with n increases, and the communication overhead is $O(n^2)$ messages [20]. Jia et al. [24] also generated partitioned data with modular addition and imposed them on a high-order polynomial coefficient. The values of the polynomial at different points are transmitted to the aggregator which finds the coefficients of the polynomial with the private key and gains the aggregation, so the scheme is under the semi-trusted model and the aggregator is trustable. In addition, the computation overhead is relatively higher when k is increasing. As every node does the x^k polynomial operation before the matrix multiplication operation, the scheme increases greatly the computation overhead.

Ohara K et al. [4] summarized the function requirements during smart metering against internal attackers: calculating billing and obtaining statistics for energy management. We follow the statistic function requirements and the spatial and temporal scenarios in References [8,23] against malicious MDMS/aggregators or dishonest users:

- (1) Spatial aggregation. A neighborhood-wide grid corresponds to a group of householders each equipped with a SG. They submit their encrypted meterings to the MDMS at a timestamp (e.g., 15 min). The latter aggregates homomorphically them before sending the aggregation to the EMS. During this aggregation, the individual data are confidential to the MDMS or the EMS.
- (2) Temporal aggregation. A single SM submits its power consumption in a series of timestamps to the MDMS for the billing purpose. In this scenario, SBS charges the householders in serial timestamps.

Throughout this paper, we refer to the building area network (BAN) region as a neighborhood, and the regional MDMS as the regional gateway (GW), and the regional SBS as the control center (CC), respectively.

The main contribution can be summarized as follows:

- (1) We design and implement a distributive, temporal and spatial aggregation scheme in the SG, in which every node sends and receives k encrypted message from k pairwise nodes distributively. The scheme provides spatial aggregation in a neighborhood at a fine-grained time scale (e.g. 15 min) and an individual temporal aggregation (e.g. monthly) in a series of timestamps for the billing purpose.
- (2) The proposed encryption scheme minimizes the computation and communication overhead by replacing the costly public key cryptography adopted in most literatures with a combination of modular addition and PRF.
- (3) The novel feature is that the masking keys are imposed on the partitioned data, and the latter are implemented by traditional modular addition. As the process of modular addition is processed by the node itself, other nodes cannot gain the true partitioned data, the masking key is only known to the pairwise nodes, and the combination ensures the confidentiality of individual data to any node including CC, aggregators, and $n-2$ nodes at most in a neighborhood.

- (4) To detect malicious aggregators or dishonest users, we propose innovatively a publicly verifiable aggregation method. By this way, any user node in a neighborhood can receive the communication flow, and verify the accuracy of local aggregation from other nodes or total aggregation from the aggregator without compromising individual fine-grained data.
- (5) The publicly available property for the aggregation also facilitates householders regulating in time its current consumption module and consumption demand in the next time period, as by comparing their own consumptions with those of other nodes and checking if there is redundant power, householders can decide to store more energy or to sell excess power to the power supplier or other nodes. The scenarios are especially very important for self-sufficient, remote places, particularly, in developing countries, which can only afford renewable energy, such as wind turbines, solar panels, and carbon-based fuels [23].

The paper is organized as follows: in Section 2, we provide related preliminaries and formalize the system and attack models. In Section 3, we introduce our proposed aggregation scheme and correctness analysis. Security notions and proof are given in Section 4, followed by performance evaluation and comparison in Section 5. The conclusion is drawn in Section 6.

2. Preliminaries and Models

For ease of reading, we summarize the main notations in the paper in Table 1.

Table 1. Notations in the scheme.

Symbol	Meaning
HSM/SM	HAN smart meter/ user/user node/sm
N	The number of users in a BAN neighborhood1
k	The number of pairwise nodes for every user
K	Keystream based on stream cipher
M	RSA modular (large prime)
$x_{(j,d)}^i$	$user_i$ partitioned data into $user_j$ at timestamp d
x_d^i	$user_i$'s data at timestamp d
$r_{(j,d)}^i$	$user_i$'s pairwise key with $user_j$ at timestamp d
$E_{(j,d)}^i$	The encrypted form of $x_{(j,d)}^i$
sk_i	The secret key between CC and every node
$ind_i[s] (s = 1, \dots, k)$	$user_i$'s pairwise nodes set in serial timestamps T
$LS(j, d)$	$user_j$'s locally spatial aggregation at timestamp d
$LT_{(j,d)}^i (j \in ind_i[s])$	$user_j$'s locally temporal aggregation for $user_i$ in T
$AT(i, T)$	$user_i$'s temporal aggregation in T
AS_d	Spatial aggregation in a neighborhood at timestamp d

2.1. Additively Homomorphic Encryption Based on The Keystream

Our security property partly comes from the stream cipher. The keystream generated from the pseudo-random function satisfies the security properties of the additively homomorphic encryption in the stream cipher. The basic idea [19] is denoted as follows:

Encryption is written as: $c = Enc_k(m + K) \bmod M$ where K is randomly generated keystream, m is the plaintext and $m, k \in [0, M - 1]$.

Decryption is described as: $Dec_k = c - K \bmod M$.

Additively homomorphic property of ciphertext are described as: $c_1 = Enc_{K1}(m_1)$ and $c_2 = Enc_{K2}(m_2)$; then, the aggregated ciphertext is expressed as: $c = c_1 + c_2 \bmod M = Enc_K(m_1 + m_2)$, where $K = K_1 + K_2 \bmod M$.

2.2. Pseudo-Random Keystream Generator—RC4

As a popular PRF generator, with secret keys between communication nodes, RC4 can generate a keystream. This secret key is pre-computed during the system initialization. As any stream cipher, the generated keystream can be used for encryption by combining it with the plaintext using bit-wise

Exclusive-Or [19]. However our scheme is to replace the XOR (Exclusive-OR) operation typically found in stream ciphers with modular addition operation (+). To generate the keystream, RC4 needs two algorithms, i.e. Key-scheduling algorithm (KSA) and Pseudo-random generation algorithm (PRGA) [5,14].

KSA: KSA is to initialize a permutation with a variable length key between 40 and 2048 bits for PRGA.

PRGA: once the permutation initialization of KSA has been completed, the stream of bits is generated using the PRGA.

Algorithm 1: Key-scheduling algorithm (KSA)

Input: $i = 0$;
 $j = 0$ // Two 8-bit index-pointers
 S // The initial key keyed with a secret key
Output: S // A permutation of all 256 possible bytes
1. **for** ($i = 0$; $i \leq 255$; $++ i$)
2. $S[i] = i$;
3. **end**
4. $k = 0$;
5. **for** ($i = 0$; $i \leq 255$; $++ i$)
6. $j = (j + S[i] + \text{key}[i \bmod \text{keylength}]) \bmod 256$;
7. $k = S[i]$;
8. $S[i] = S[j]$;
9. $S[j] = k$;
10. **end**

Algorithm 2: Pseudo-random generation algorithm (PRGA)

Input: $i = 0$;
 $j = 0$ // Two 8-bit index-pointers
Output: Z // Pseudo-random keystream
1. $k = 0$;
2. **for** ($i = 0$; $i \leq 255$; $++ i$)
3. $i = (i + 1) \bmod 256$;
4. $j = (j + S[i]) \bmod 256$;
5. $k = S[i]$;
6. $S[i] = S[j]$;
7. $S[j] = k$;
8. $Z = S[(S[i] + S[j]) \bmod 256]$;
9. **end**

2.3. System Model

In our system model, we consider a typical SG communication architecture [8,9,11,15–17], as shown in Figure 1. It is based on the SG network model presented from the National Institute of Standards and Technology (NIST) and consists of six domains, i.e., the power plant, the transmission domain, the distribution domain and a CC, a residential GW, and the user domain. We mainly focus on how to report and aggregate the users' privacy-preserving data into the CC. Hence, the system model divides especially the BAN into numbers of Household area network (HAN) equipped with a SG and every BAN includes a GW and numbers of users.

CC: It acts as the SBS and EMS in reality. It needs to monitor the actual data on how much power is consumed at which timestamp in one BAN (neighborhood), how much power should be reserved for the next time period, and cumulative consumption for individual billing on a monthly basis, and how much power is being distributed to a specified neighborhood. In the paper, it is curious about the individual fine-grained data and may attempt to it as far as possible by all available resources, so it is assumed a semi-trusted entity.

GW: A powerful entity, acting as the local MDMS, represents a locality (e.g., a region within a building) is responsible for aggregating real-time spatial data in a neighborhood and individual temporal data in a series of timestamps, and then transmitting the aggregation to the CC. The employment of GW relieves CC of aggregation and reducing largely the communication latency. However, the cost that potentially malicious attacks done to users or power suppliers is unignorable, as discussed earlier. We assume it is a malicious entity here. A **BAN GW** represents a locality (e.g., a region within a neighborhood). For facilitating the communication between BAN GW and CC, WiMax and other broadband wireless technologies can be adopted. We consider a scenario that one BAN neighborhood covers a hundred or more HANs, so the longest distance from the BAN GW to a HAN is more than a hundred miles, so WiMax maybe more suitable for this kind of distance communication.

Household Smart Meter (HSM): A bidirectional communication entity deployed at householders' premises. The modern SM is given a certain level of autonomy via trusted elements and the ability to collect, store, aggregate, and encrypt the usage data. Hence it has two interfaces—one interface is for reading power of householders and the other one acts as a communication GW. Even if we assume SM is tamper-resistant, it is not powerful as a GW, so it may be vulnerable to be compromised by the GW to infer the object users' data.

Figure 1. System model under consideration.

2.4. Communication Model

As can be seen in the Figure 1, all SMs connect each other in a neighborhood by WiFi technique, which constructs public verifiable foundation. Each user would select randomly k pairwise nodes in one round and can ensure that if $user_i$ chooses $user_j$, then $user_j$ chooses $user_i$ and the keys between them are opposite mutually. The value k as a security parameter can take any value from 2 to n , and depend on the specific application circumstance. The higher the value of k is, the higher the complexity is, and vice versa, and the scheme is more vulnerable to be attacked.

2.5. Data Model

Let x_d^i be the meter reading of the i th ($1 \leq i \leq N$) user node at the d th ($1 \leq d \leq T$) fine-grained timestamp, where N is the number of user in a BAN (a neighborhood-wide grid), and T is a billing period. At each fine-grained time index d , a neighborhood grid (over the entire BAN) spatially aggregated utility usage can be expressed as:

$$AS(d) = \sum_{i=1}^n x_d^i; \quad d = 1, 2, \dots, T \quad (1)$$

At the end of a billing period ($d = T$), a temporally aggregated utility usage data for the i th user is expressed as:

$$AT(i, T) = \sum_{d=1}^T x_d^i; i = 1 \text{ to } N \quad (2)$$

2.6. Security Requirement and Attack Model

Within the system model, there are four types of actors involved in the meter data reporting process: the i th user (self), other users in the same neighborhood (BAN), the GW, and the CC. The CC requires the spatially aggregated fine-grained neighborhood usage data to optimize power delivery efficiency and the temporally aggregated user-specific utility usage data for the billing purpose. Hence, we stipulate the following security/privacy requirements:

Requirement R1. Fine-grained, individual utility data are private and should not be disclosed to CC, GW, or other users.

Requirement R2. Temporal aggregation for an individual user and spatial aggregation in one neighborhood cannot be tampered by the malicious aggregator or other internal nodes. For this, we envision a secure and reliable communication model comprising a verifiable publically method, which is customized to the correctness verification of the aggregation value of SG.

For this, our attack model is based on the malicious aggregator who attempts to tamper the aggregation value in a neighborhood and the billing value for individual users, or infers fine-grained meterings of the individual user by colluding with other $n-2$ compromised nodes at most. Following the above security requirements, different compositions of the attackers and actions may be grouped into the following attack types:

(1). External attack

External attackers may compromise the meterings of the object users by eavesdropping the communication flow between communication nodes through various eavesdropping malware.

(2). Malicious attack

False aggregation report. The aggregator may alter or drop maliciously any individual data, or tamper the aggregation data to the CC; any malicious user node may provide false local aggregation to the GW.

Collusion with compromised nodes. The aggregator may collude with compromised users to attempt to infer the uncompromised users' data.

(3). Semi-trustable internal attack

The curious CC or any user node can also acquire data through the public communication flow, such as the message from the user node to the GW or from the GW to the CC. They may infer the object user's fine-grained data by the public communication flow.

An attack is an arrangement that enables unauthorized parties to gain access to private data or to tamper secured data (even by the user itself) without being detected. In this work, we assume the SMs are tamper-resistant [7,20,23], and can perform the measurement and reporting operations normally, but do not exclude the possibility of tampering with local aggregation values by itself.

3. Proposed Scheme

3.1. Initialization Phase

3.1.1. Initializing Pairwise Number k and Session Key

For every billing period, the CC generates randomly the pairwise number for every node in one neighborhood denoted as k , and broadcasts it to all SMs.

We generate session keys between every node with the computational DH key exchange protocol as the initial key in RC4 to generate the keystream between pairwise nodes. Once one node joins a

neighborhood size of n , it generates itself one DH public key $g^a \pmod{M}$ and remains the secret key a , M are DH parameters, and then broadcasts the public key. By this Computational Diffie-Hellman CDH exchange key, any two pairwise nodes can identify their session key formed as g^{ab} .

3.1.2. Modular Addition

The $user_i$ partitions its own data x_d^i into k partitions denoted as $x_{(ind_i[j],d)}^i$ ($1 \leq j \leq k$) and sends them to every pairwise node. However, the partitioned data can be easily guessed, especially with brute search, as the consumption value at every timeslot is very small. For this, we impose extra noise (masking keys) which is only known by pairwise nodes themselves on the partitioned data to further secure the individual data.

3.1.3. Noise Addition

Masking keys, as extra noise, are generated by pairwise nodes with PRF at every timestamp. The PRF can be implemented with RC4, the specific process can be referred to the Section 2.2.

3.2. Encryption and Aggregation

3.2.1. Data Encryption

(1). Partition of individual data

Each node randomly partitions its individual data into k partitions and sends them to k pairwise nodes along with the masking keys. The partition form is as follows:

$$x_d^i = \sum_{j \in ind_i[s]} x_{(j,d)}^i \quad (1 \leq s \leq k) \quad (3)$$

(2). Generation of pairwise nodes and masking keys

For any node, it chooses randomly any k nodes in one round as its pairing nodes such that if $user_i$ selects $user_j$, then $user_j$ also selects $user_i$. With the session key between them, the two pairwise nodes generate a common key r from RC4; $user_i$ adds $r_{(j,d)}^i$ to $x_{(j,d)}^i$, and $user_j$ adds $r_{(i,d)}^j$ which satisfies:

$$r_{(i,d)}^j = -r_{(j,d)}^i \quad (i \in ind_j[s]; j \in ind_i[s]) \quad (4)$$

For $user_i$, the generated noise set at the timestamp d can be denoted as $r_{(ind_i[s],d)}^i$ ($s = 1, 2, \dots, k$). Note that in order to facilitate the temporal aggregation, the pairwise key generated by an SM at the T^{th} timestamp should satisfy the following equation:

$$r_{(i,d)}^j = -r_{(j,d)}^i \quad (i \in ind_j[s]; j \in ind_i[s]) \quad (5)$$

(3). Encryption process

At the timestamp d , $user_i$ adds the pairwise noise to the partitioned data to generate the encrypted message $E_{(j,d)}^i = x_{(j,d)}^i + r_{(j,d)}^i$ ($j \in ind_i[s]$) to k pairwise nodes separately as well as receiving the encrypted message they sent. The Figure 2 illustrates an example for spatial and temporal aggregation among pairwise users in multi-region groups.

Figure 2. Example multi-region spatial and temporal aggregation.

For any SM node j , it will store the encrypted data sent from one of its pairwise node i in a series of T timestamps in the form of matrix as follows:

$$\begin{bmatrix} E_{(j,1)}^1 & E_{(j,1)}^2 & \cdots & E_{(j,1)}^i & \cdots & E_{(j,1)}^n \\ E_{(j,2)}^1 & E_{(j,2)}^2 & \cdots & E_{(j,2)}^i & \cdots & E_{(j,2)}^n \\ \vdots & \vdots & & \vdots & & \vdots \\ E_{(j,T)}^1 & E_{(j,T)}^2 & \cdots & E_{(j,T)}^i & \cdots & E_{(j,T)}^n \end{bmatrix} \begin{pmatrix} E_{(j,1)}^i = (x_{(j,d)}^i + r_{(j,d)}^i) & j \in \text{ind}_i[s] \\ E_{(j,d)}^i = 0 & j \notin \text{ind}_i[s] \end{pmatrix} \quad (6)$$

3.2.2. Storage and Aggregation

(1). Spatial Aggregation

Once receiving encrypted data at timeslot d from all pairwise nodes, $user_i$ aggregates them and generates the local spatial data $LS(i, d)$ as follows:

$$LS(i, d) = \sum_{j \in \text{ind}_i[s]}^n (x_{(i,d)}^j + r_{(i,d)}^j) \bmod M \quad (1 \leq s \leq k) \quad (7)$$

Every user sends the local spatial aggregation formed as $LS(i, d)$ to the GW at every timestamp.

Once receiving the locally spatial aggregation $LS(i, d)$ from the pairwise nodes, the GW adds them up together and the pairwise keys cancel out. The total spatial aggregation is denoted as:

$$AS_d = \sum_{i=1}^n LS(i, d) \bmod M \quad (8)$$

(2). Temporal aggregation

Every user node receives the encrypted data from its pairwise nodes and stores it as a matrix of T rows and n columns formed as Equation (6).

In every billing period T , the user node aggregates every column in the Equation (6) into locally temporal aggregation after the pairwise keys cancel out. The locally temporal aggregation form is as follows:

$$\begin{aligned} LT_{(j,T)}^i &= \sum_{d=1}^T E_{(j,d)}^i \bmod M (j \in ind_i[s]) \\ &= \sum_{d=1}^T (x_{(j,d)}^i + r_{(j,d)}^i) \bmod M \end{aligned} \quad (9)$$

Once the CC issues the temporal aggregation request for $user_i$ to the GW, the pairwise nodes of $user_i$ would report its local temporal aggregation $LT_{(j,T)}^i$ to the GW.

The GW aggregates them into the temporal aggregation and transmits it to the CC; the aggregation process is as follows:

$$AT(i, T) = \sum_{j \in ind_i[s]} LT_{(j,T)}^i \bmod M (1 \leq s \leq k) \quad (10)$$

We assume $j \in ind_i[s]; i \in ind_j[s]$.

Figure 3 shows the communication process between the pairwise nodes and GW at the timestamp d .

Figure 3. Communication process data between the pairwise nodes and GW.

3.2.3. Decryption Process

In this way, the aggregation process is actually the decryption process, in which the random keys cancel out and individual consumption in a billing period or the spatial aggregation in a neighborhood is revealed. Hence the combination of simple modular addition with noise addition reduces the costly encryption and decryption operation in public key cryptography.

3.3. Correctness Analysis

Now we prove the correctness of our encryption scheme in terms of spatial and temporal aggregation:

3.3.1. Spatial Aggregation

$$\begin{aligned}
 AS_d &= \sum_{i=1}^n LS(i, d) \bmod M \\
 &= \sum_{i=1}^n \sum_{j \in ind_i[s]} (x_{(i,d)}^j + r_{(i,d)}^j) \bmod M \quad (1 \leq s \leq k) \\
 &= \left(\sum_{i=1}^n \sum_{j \in ind_i[s]} x_{(i,d)}^j + \sum_{i=1}^n \sum_{j \in ind_i[s]} r_{(i,d)}^j \right) \bmod M \\
 &= \left(\sum_{j=1}^n \sum_{i \in ind_j[s]} x_{(i,d)}^j + \frac{1}{2} \left(\sum_{i=1}^n \sum_{j \in ind_i[s]} r_{(i,d)}^j + \sum_{j=1}^n \sum_{i \in ind_j[s]} r_{(i,d)}^j \right) \right) \bmod M \\
 &= \left(\sum_{j=1}^n x_d^j + \frac{1}{2} \left(\sum_{i=1}^n \sum_{j \in ind_i[s]} r_{(i,d)}^j - \sum_{i=1}^n \sum_{j \in ind_i[s]} r_{(i,d)}^j \right) \right) \bmod M \\
 &= \sum_{j=1}^n x_d^j \bmod M
 \end{aligned} \tag{11}$$

We prove the correctness of our spatial aggregation by permuting the row and column of data matrix formed as Figure 2. Equation (11) shows that the spatial aggregation in a neighborhood equals to the sum of locally spatial aggregation, i.e., the sum of individual data.

3.3.2. Temporal Aggregation

$$\begin{aligned}
 AT(i, T) &= \sum_{j \in ind_i[s]} LT_{(j,d)}^i \bmod M \quad (1 \leq s \leq k) \\
 &= \sum_{j \in ind_i[s]} \sum_{d=1}^T E_{(j,d)}^i \bmod M \\
 &= \sum_{d=1}^T \sum_{j \in ind_i[s]} (x_{(j,d)}^i + r_{(j,d)}^i) \bmod M \\
 &= \sum_{d=1}^T x_d^i \bmod M
 \end{aligned} \tag{12}$$

Equation (12) shows that the temporal aggregation for one user node equals to the sum of local temporal aggregation from its pairwise nodes, i.e., the sum of its individual data in a series of timestamps T . It proves further the correctness of our temporal aggregation.

4. Security Notions

4.1. Security Proof

In this section, we mainly elaborate the security properties of our scheme. In particular, based on the security requirement and attack model discussed in Section 2.6, we prove our scheme can ensure the confidentiality of fine-grained meterings for an individual user and the aggregation integrity that the local aggregation, and total aggregation cannot tampered by malicious individual user nodes or the aggregator.

We firstly construct the Individual Metering Indistinguishable (IMI) security game to represent the adversary's actions.

Definition 1. (IMI security game).

Setup: the challenger runs the initialization algorithm and first initializes a group of size n , then generates the system parameter k to the adversary.

Queries: the adversary can not only capture meters' encrypted report but also acquire the encryption and compromise queries until meeting the constraints.

Encrypt: The adversary A chooses user i and specifies x_d^i to ask for the ciphertext. The challenger returns it the ciphertext $E(x_d^i)$.

Compromise: The adversary A specifies an integer $q \in \{0, 1, \dots, n\}$. If $q = 0$, the challenger returns the adversary the aggregator' capability, else returns user q 's message.

Challenge. We denote with $\{\bar{C}\}$ the set of the uncompromised users. The adversary selects randomly two meterings $x_d^{i_0}$ and $x_d^{i_1}$ ($i \in \{\bar{C}\}$) at the timestamp d . The challenger flips a random bit $b \in \{0, 1\}$ uniformly and returns the adversary $E(x_d^{i_b})$.

Guess: The adversary outputs a guess $b' \in \{0, 1\}$, and A wins if $b = b'$ with unignorable advantage.

Definition 2. (IMI security)

The proposed temporal and spatial aggregation scheme is IMI if no probabilistic polynomial-time adversaries A have more than an ignorable advantage in the IMI security game. The ignorable function for A is as follows:

$$Adv_A = |\Pr[b = b'] - \frac{1}{2}| = 0 \quad (13)$$

Theorem 1. The proposed encryption scheme is IMI.

The intuition behind the theorem is any adversary cannot distinguish the encrypted individual metering and the scheme cannot leak any individual user consumption at the d^{th} timestamp.

Proof:

Setup: The challenger initiates the whole system. The challenger generates a group of scale n and pairs number k , and then gives the parameters (n, k) to the adversary.

Queries:

(1). Spatial aggregation

Encrypt: A issues the encryption query with (i, d, x_d^i) to the challenger. The challenger generates the pairwise key $r_{(j,d)}^i$ ($j \in ind_i[s]$) between the pairwise nodes, and imposes it on the randomly partitioned data $x_{(j,d)}^i$ ($j \in ind_i[s]$) to generate the encrypted measure formed as $E(x_d^i) = x_{(j,d)}^i + r_{(j,d)}^i \bmod M$ ($j \in ind_i[s]$).

Compromise: A may compromise the aggregator or up to $n-1$ users in any pairwise set in order to acquire more messages for object users. However, the compromise will encounter restriction when meeting with uncompromised users.

Challenge. For simplifying the proof process and not losing the generalization, we consider the extreme circumstance that $|\bar{C}| = 2$. If the theorem holds for this circumstance, then it holds for $|\bar{C}| > 2$. We assume the user j is the only uncompromised user in $ind_i[s]$ ($1 \leq s \leq k$). The adversary selects the two meterings and gives $(i, j, d, x_d^{i_0}, x_d^{i_1})$ to the challenger, the challenger flips a random bit $b \in \{0, 1\}$ uniformly and returns the adversary $E(x_d^i)$ when $b = 0$, and returns $E(x_d^j)$ when $b = 1$, and then

$$\begin{aligned} E(x_d^i) &= \sum_{l \in ind_i[s]} (x_{(l,d)}^i + r_{(l,d)}^i) \bmod M \quad (1 \leq s \leq k) \\ &= (x_{(j,d)}^i + r_{(j,d)}^i) + \sum_{c \in \{ind_i[s] - j\}} (x_{(c,d)}^i + r_{(c,d)}^i) \bmod M \end{aligned} \quad (14)$$

$$\begin{aligned} E(x_d^j) &= \sum_{l \in ind_j[s]} (x_{(l,d)}^j + r_{(l,d)}^j) \bmod M \quad (1 \leq s \leq k) \\ &= (x_{(i,d)}^j + r_{(i,d)}^j) + \sum_{c \in \{ind_j[s] - i\}} (x_{(c,d)}^j + r_{(c,d)}^j) \bmod M \end{aligned} \quad (15)$$

In the Equations (14) and (15), the adversary A cannot solve the two equations at the d^{th} timestamp and gain the exact $x_{(j,d)}^i$ and even if he knows $r_{(i,d)}^j = -r_{(j,d)}^i$, as the two equations have three unknown variables, so it is more impossible for A to acquire x_d^i and x_d^j which ensures the scheme's security.

(2). Temporal aggregation

$$\begin{aligned}
 E\left(\sum_{d=1}^T x_d^i\right) &= \sum_{d=1}^T \sum_{l \in \text{ind}_i[s]} (x_{(l,d)}^i + r_{(l,d)}^i) \bmod M \quad (1 \leq s \leq k) \\
 &= \left(\sum_{d=1}^T (x_{(j,d)}^i + r_{(j,d)}^i) + \sum_{d=1}^T \sum_{c \in \{\text{ind}_i[s]-j\}} (x_{(c,d)}^i + r_{(c,d)}^i) \right) \bmod M \\
 &= ((x_{(j,T)}^i + r_{(j,T)}^i) + \sum_{d=1}^{T-1} (x_{(j,d)}^i + r_{(j,d)}^i) + \sum_{d=1}^T \sum_{c \in \{\text{ind}_i[s]-j\}} (x_{(c,d)}^i + r_{(c,d)}^i)) \bmod M \\
 &= (x_{(j,T)}^i + \sum_{d=1}^{T-1} x_{(j,d)}^i + \sum_{d=1}^T \sum_{c \in \{\text{ind}_i[s]-j\}} (x_{(c,d)}^i + r_{(c,d)}^i)) \bmod M
 \end{aligned} \tag{16}$$

$$E\left(\sum_{d=1}^T x_d^j\right) = (x_{(i,T)}^j + \sum_{d=1}^{T-1} x_{(i,d)}^j + \sum_{d=1}^T \sum_{c \in \{\text{ind}_j[s]-i\}} (x_{(c,d)}^j + r_{(c,d)}^j)) \bmod M \tag{17}$$

□

In the Equations (16) and (17), the two equations with four unknown variables make the adversary A impossible to acquire $x_{(j,d)}^i$ or $x_{(i,d)}^j$.

Hence, the encrypted aggregation method can ensure the individual, fine-grained meterings indistinguishable security as long as there is at least one uncompromised user in its pairwise set. Our security properties are based on the randomness of modular addition and stream cipher which is used to blind the individual meterings.

4.2. Security Analysis

We can prove that our proposed solution will withstand the other attacks discussed in Section 2.6 and ensure the integrity of the aggregated data, whether total aggregation or local aggregation.

(1). Eavesdropping resistance

Our proposed scheme supports the openness of communication flow. Whether it is the internal node with access to the communication flow in a community or the external eavesdropper, they can only get the encrypted individual data $(x_{(j,d)}^i + r_{(j,d)}^i)$, local aggregation value $(LS(i,d), LT_{(j,d)}^i)$ or total aggregation value $(AS(d), AT(i,T))$ sent by GW to CC. However, all of them can not obtain the fine-grained data. We have proved that even if all but one node is compromised, object metering still cannot be leaked. Hence, the proposed encryption method satisfies the security requirement R1.

(2). False command from the GW

The GW attempts to obtain object user's meterings by issuing false billing commands in the name of CC, even if he cannot compromise its pairwise nodes. He tries to obtain valuable information from them at any timestamp. However, even so, he can only get the indistinguishable, individual meterings, due to the Equations (14)–(17).

We cannot exclude the possibility that all pairwise keys of $user_i$ at a timestamp are all compromised nodes by the malicious aggregator or external attackers. In this case, the object $user_i$'s privacy is exposed. That is the $user_i$ does not select any one honest node, then the probability is $1 - (\frac{k}{n-1})^{n-1-|c|}$. Obviously, the larger the value of $|c|$ is, the smaller the value of k is, and the bigger the probability is. We improve the probability as much as possible and assume $n = 1000$, $k = 30$, and $|c| = 500$ (50% nodes are compromised), and then the probability is 2.47×10^{-7} , so much small probability implies it

is almost impossible that one user does not select any one honest node in one timestamp. Even if we fix a bigger pairs period $T = 1$ month, then we would have to cost 38.51 years to acquire individual data.

4.3. Publicly Verifiable Property

The security requirement R2 given earlier needs to be satisfied with the publicly verifiable property. We provide the public communication flow between nodes in a neighborhood is to ensure the integrity of aggregation data. Any internal node in the community can verify publicly the accuracy of the local aggregation from other nodes and the total aggregation from the GW without compromising the individual fine-grained data. The special public verification process comprises two parties:

4.3.1. Spatial Verification

Based on the public communication flow, any node in the neighborhood can gain the encrypted message formed as $x_{(j,d)}^i + r_{(j,d)}^i$ from the pairwise nodes, and compute its local aggregation formed as $LS(i,d)$ and $LT_{(j,d)}^i$, and thus the total aggregation AS_d and $AT(i,T)$ for the neighborhood can be computed and compared with the reported result from the GW. If the result is questionable, the user can report directly to the CC. With such a supervision, the CC can detect the fraudulent profile of the malicious GW.

4.3.2. Temporal Verification

The public verification method to the spatial aggregation is equally effective to the temporal verification. For any node, one of its pairwise nodes in the neighborhood gain its encrypted message formed as $x_{(j,d)}^i + r_{(j,d)}^i$ in a billing period before computing its local temporal aggregation, and thus its total temporal aggregation is computed and verified by summing up local temporal aggregations from all its pairwise nodes.

Thus, the billing user itself or any user node can verify the accuracy of the billing from the GW without revealing individual fine-grained data. Hence, they can detect if there is a malicious and fraudulent profile of the malicious GW and reports it to the CC in time.

5. Performance Evaluation

We evaluate the performance of the proposed aggregation scheme to assess the overheads. The performance metrics used in our empirical evaluation are defined as follows:

- (1) Computation overhead: node's runtime of the proposed scheme in terms of spatial and temporal aggregation.
- (2) Communication overhead: the size of a message transmitted between the nodes and GW (number of bits).
- (3) Security parameter k : we analyze the impact of the different value of k on the two overheads.

We compare these results against several existing works [23,24] using performance metrics based on Friendly ARM [25] and the library in [17]. By comparison with them, we intend to illustrate our computing and communication advantages in terms of the combination of PRF and modular addition methods adopted, respectively, in the scheme [23] and [24]. Each experiment consists of 50 independent trials and the averaged results of these trials are reported. The computation time required for these tasks is listed in Table 2.

Table 2. Average time for functions.

Notations	Descriptions	Time Cost
C_{add}	Addition	≈ 0.038 ms
C_{hash}	Hash (100 randoms)	≈ 0.85 ms
C_{mul}	Multiplication	≈ 0.013 ms
C_{enc}	Homomorphic encryption	≈ 2.7 ms
C_{dec}	Homomorphic decryption	≈ 0.61 ms
C_{ma}	Hash/Modular addition	≈ 0.0023 ms
C_{prf}	Pseudorandom function	≈ 0.074 ms

We fix the number of users at 1 million; the number of C is 10; the number of GW ranges from 1 to 20. Let n denotes a possible number of users in a group, and it ranges from 1 to 5000. We present the impact of a different number of users in the GW and a different value of k (ranging from 1 to 100) on the performance. We also assume, for simplicity, that all SMs can be functioning normally.

5.1. Computation Overhead

(1). Spatial aggregation

Let C_{ma} and C_{prf} denote respectively the cost of Modular addition operation and keys generation operation with PRF, respectively let C_{add} and C_{mul} denote the cost of addition and multiplication operation respectively, and C_{enc} and C_{dec} denote the cost of homomorphic encryption and decryption operation respectively.

In our spatial aggregation scheme, for every node, partitioning individual data into k partitions costs one C_{ma} ; generating k pairwise keys costs $k \cdot C_{prf}$; receiving k encrypted messages and adding them up cost $k \cdot C_{add}$, then the computation overhead per node is $C_{ma} + k \cdot C_{prf} + k \cdot C_{add}$ and the total computation overhead per aggregator is $(n-1) \cdot C_{add}$ for aggregating data from n nodes.

In Erkin et al.'s scheme [23], at the d^{th} time step, every hash function cost is C_{hash} , k masking random keys cost is $k \cdot C_{prf}$ and computing total masking keys cost is $2k \cdot C_{add}$, and then encrypting individual data cost is C_{enc} , so the total computation overhead is $C_{hash} + k \cdot C_{prf} + 2k \cdot C_{add} + C_{enc}$.

In Jia et al.'s scheme [24], at the d^{th} time step, the additive secret sharing cost is C_{ss} , k hash functions cost is $k \cdot C_{hash}$, and then k -order polynomial operation is x^k and k matrix multiplication operations cost is $(k^2 + 2k) \cdot C_{mul}$, so the total computation overhead is: $C_{ss} + k \cdot C_{hash} + (k^2 + 2k) \cdot C_{mul}$.

We provide the individual spatial computation overhead comparison in Table 3.

Table 3. Individual spatial computation overhead comparison (msec).

Scheme	Computation Overhead Per Smart Meter
Scheme in [23]	$C_{hash} + k \cdot C_{prf} + 2k \cdot C_{add} + C_{enc}$
Scheme in [24]	$C_{ss} + k \cdot C_{hash} + (k^2 + 2k) \cdot C_{mul}$
Our scheme	$C_{ma} + k \cdot C_{prf} + k \cdot C_{add}$

As described in the related work, the scheme in Reference [23] sets all nodes as communication nodes instead of selecting a limited number of communication nodes as in ours and [22]; however, for convenient comparison, we assume that k communication nodes are selected, which is on the same experiment platform as ours and the scheme in [23]. Even under such relaxation, we can still prove ours is superior in terms of computation and communication cost through the following performance evaluation.

The Figure 4 plots the comparison of spatial computation overhead between our scheme and the schemes in References [23,24] with the value of k increasing. The Figure 4 shows that the three schemes' computation overheads all increase with the value of k increasing, the computation overhead in Reference [23] and ours are lower compared with the scheme in References [24], in which polynomial

operation x^k and k matrix multiplication operations generate too much computation overhead with k growing, it has more cost significantly than ours and Erkin et al.'s scheme [23], ours is lower slightly than the scheme in [23], and both of them are close to $O(k) \cdot C_{prf}$.

Figure 4. Variety of spatial computation overhead per node with the value k .

(2). Temporal aggregation

In the proposed scheme, each node chooses the same nodes every billing period to satisfy with the Equation (5), so total temporal computation overhead in T serial time slots for every node is $T \cdot (k \cdot C_{prf} + k \cdot C_{add} + C_{ma}) + T \cdot C_{add}$.

In Erkin et al.'s scheme [23], each node sends T fine-grained utility readings in each of the T time steps, so the overhead per node is $T \cdot (C_{hash} + k \cdot C_{prf} + 2k \cdot C_{add} + C_{enc}) + T \cdot C_{mul}$. In fact, the temporal aggregation overhead of the scheme in Reference [23] is higher than it, as with the modification of Paillier encryption, spatial and temporal aggregations are not being synchronized. To compensate the lack, every user must add an additional random key $R_{(i,T+1)} = \frac{r^n}{\prod_{d=1}^T h_d^{R_{(i,d)}}}$ at T^{th} timestamp, which costs much overhead. However, our scheme has no extra cost and the third party's involvement.

We set the fine-grained reporting interval to be 15 minutes, and billing period $T = 2880$ (roughly one month). Figure 5 plots the comparison of two schemes in terms of temporal computation overhead in every billing period for k ranging from 0 to 50. From Figure 5, we can see the temporal computation overhead per node grows with the increasing of k value in two schemes; however, our proposed scheme increases slightly compared with the scheme in References [23], as the latter costs much overhead on Paillier encryption, while our scheme achieves the same privacy protection effect as the asymmetric encryption with simple and low-cost modular addition.

Figure 5. Variety of temporal computation overhead per node with the value k .

5.2. Communication Overhead

We assume the format of a packet is the same as that in TinyOS [26]. The timestamp occupies 128 bits. The sizes of prime numbers p , and q needed in the Paillier encryption are 512 bits each. The size of elements in Z_n^* is 1024 bits. We further assume the plaintext data occupies 32 bits, then random from stream cipher occupies the same byte width with the plaintext data, and Paillier encryption occupies 4096 bits, while the hash function with timestamp occupies 256 bits.

For simplicity, we denote $\{|X|, |R|, |E|, |H|\}$ as the plaintext data size, masking random key (noise) size, Paillier encryption size, and the size of hash function random.

5.2.1. Spatial Communication Overhead Per Node

To generate the spatial aggregation, every node sends the local aggregation to the GW after adding up the encrypted message from all k pairs. The data sent per node can be denoted as $\{LS(i, d) \| t\}$, the size is $|X| + k \cdot |R| + 128$ bits (a partitioned part size is $\frac{|x|}{k}$ bits, k partitions take $|x|$ bits; a noise key takes $|R|$ bits, and then k noise keys take $k \cdot |R|$ bits), so the total packet size is $|x| + k \cdot |R| + 128$ bits.

For the scheme in Reference [23], the spatial aggregation packet per node is in the form as $\{E \| H \| R \| t\}$, its size is $\{|E| + |X| + k \cdot |R| + |H| + 128\}$ bits.

Every user node in Reference [24] generates k results, the data is in the form of $\{(y_1 \| y_2 \| \dots \| y_k) \| t\}$, in which y_i involves the computation of data sharing and hash random value, so its size is $K \cdot (K \cdot |H| + |X|/k + 128)$ bits.

We provide the individual spatial communication overhead comparison in Table 4.

Table 4. Individual spatial communication overhead comparison (bits).

Scheme	Computation Overhead Per Smart Meter
Scheme in [23]	$ E + X + k \cdot R + H + 128$
Scheme in [24]	$K \cdot (K \cdot H + X /k + 128)$
Our scheme	$ X + k \cdot R + 128$

We plot the individual communication overhead comparison between our scheme and the other two schemes [23,24] during spatial aggregation in the Figure 6. We can see clearly the three schemes' individual overhead all grow with the increasing of k value. The packet width per node in the scheme in Reference [24] grows significantly than the other two schemes, especially when k value is relatively higher, and communication overhead closes to $O(k^2)$, due to the x^k polynomial operation per

node before the matrix multiplication operation. Our scheme's growth rate is close to the scheme in Reference [23], which is higher always slightly higher than ours, due to the relatively higher public key encryption width.

Figure 6. Variety of spatial communication overhead per node with the value k .

5.2.2. Temporal Communication Overhead Per Node

Figure 7 shows the comparison result of ours and the scheme [23] in terms of temporal communication overhead per node when k ranges from 0 to 600, and T ranges from 0 to 6000 mins.

Figure 7. Variety of temporal communication overhead per node with the values k and T .

In Figure 7, our scheme reduces significantly the packet size sent per node to almost three orders of magnitude than the scheme [23], due to the high overhead of public key encryption. During temporal aggregation, if the process of exchanging random between communication nodes is ignorable, then every node sends its serial encrypted packet formed as $\{E \parallel H \parallel R \parallel t\} (1 \leq t \leq T)$ to the aggregator, so the packet size is $T \cdot (|E| + |X| + k \cdot |R| + |H| + 128)$ bits, while in our scheme, one node's temporal aggregation is computed synchronously before being reported to the aggregator by k communication nodes, and they send the local temporal aggregation packet size of $|x| + k \cdot |R| + 128$ bits to the

aggregator every T timeslot, so aggregating one node's temporal consumption in T serial time slots costs $k \cdot (|x| + k \cdot |R| + 128)$ bits. Hence, when $k \ll T$, ours overhead is always lower significantly lower than the scheme in Reference [23]. Just as the description above, we shorten the number of the communication nodes in Reference [23] into k , and the performance evaluation shows the proposed collection of modular addition and masking keys from PRF saves much computation and communication overhead compared with traditional public key encryption without compromising individual privacy.

6. Conclusions

In the paper, we resolved three issues about privacy-protection aggregation of smart metering customized to the SG. Firstly, the combination of simple modular addition and PRF we designed serves the same effect as the other most related works with lower overhead, namely fending off maliciously internal attacks without compromising individual fine-grained data. Secondly, we proposed innovatively a publicly verifiable platform, by which, every node in a neighborhood can verify local aggregation from every node and total aggregation from the GW and detect the fraudulent profiles from maliciously internal nodes or dishonest user nodes. Thirdly, every node chooses randomly k nodes rather than all nodes as pairwise nodes to communicate, which saves significantly communication and computation overhead, and the independence of the number of users provides scalability and high efficiency under the circumstance of SG big data. From the performance evaluation shows that the proposed scheme is applicable for the security and privacy protection of SG and has practical significance.

Author Contributions: L.Z. and J.Z. designed the hierarchical architecture model, attack models, communication models, and encryption methods together; J.Z. optimized the communication models, and L.Z. wrote the paper.

Funding: This research was funded by National Natural Science Foundation of China (NSFC) (2017–2020, No. 51679058).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wang, W.; Lu, Z. Cyber security in the Smart Grid: Survey and challenges. *Comput. Netw.* **2013**, *57*, 1344–1371. [CrossRef]
2. Ambrosin, M.; Hosseini, H.; Mandal, K. Despicable meter: Anonymous and fine-grained metering data reporting with dishonest meters. In Proceedings of the 2016 IEEE Conference on Communications and Network Security (CNS 2016), Philadelphia, PA, USA, 17–19 October 2016; pp. 163–171.
3. Krebs, B. FBI: Smart Meter Hacks Likely to Spread. 2013. Available online: <http://krebsonsecurity.com/2012/04/fbi-smart-meter-hackslikely-to-spread/> (accessed on 07 April 2012).
4. Ohara, K.; Sakai, Y.; Yoshida, F.; Iwamoto, M.; Ohta, K. Privacy-preserving smart metering with verifiability for both billing and energy management. In Proceedings of the 2nd ACM Workshop on ASIA Public-Key Cryptography (ASIAPKC'14), Kyoto, Japan, 3–6 June 2014; pp. 23–32.
5. Lincoln, K.; Philip, K.; Christopher, M. The use of RC4 encryption for smart meters. In Proceedings of the 2014 International Conference on Sustainable Research and Innovation, Nairobi, Kenya, 7–9 May 2014; pp. 58–62.
6. Lu, R.; Liang, X.; Li, X.; Lin, X.; Shen, X. EPPA: An efficient and privacy-preserving aggregation scheme for secure smart grid communications. *IEEE Trans. Parallel Distrib. Syst.* **2012**, *23*, 1621–1631. [CrossRef]
7. Garcia, F.D.; Jacobs, B. Privacy-friendly energy-metering via homomorphic encryption. *IEEE Trans. Parallel Distrib.* **2010**, *6710*, 226–238. [CrossRef]
8. Wang, X.; Mu, Y.; Chen, R. An efficient privacy-preserving aggregation and billing protocol for smart grid. *Secur. Commun. Netw.* **2016**, *9*, 4536–4547. [CrossRef]
9. Chen, L.; Lu, R.; Cao, Z. PDAFT: A privacy-preserving data aggregation scheme with fault tolerance for smart grid communications. *Peer-to-Peer Netw. Appl.* **2015**, *8*, 1122–1132. [CrossRef]

10. He, D.; Kumar, N.; Zeadally, S. Efficient and Privacy-Preserving Data Aggregation Scheme for Smart Grid against Internal Adversaries. *IEEE Trans. Smart Grid* **2017**, *8*, 2411–2419. [[CrossRef](#)]
11. Bao, H.; Lu, R. A New Differentially Private Data Aggregation with Fault Tolerance for Smart Grid Communications. *IEEE Internet Things J.* **2015**, *2*, 248–258. [[CrossRef](#)]
12. Kursawe, K.; Danezis, G.; Kohlweiss, M. Privacy-friendly aggregation for the smart-grid. In Proceedings of the International Symposium on Privacy Enhancing Technologies Symposium, Cambridge, UK, 27–29 July 2011; pp. 175–191.
13. Shi, Z.; Sun, R.; Lu RChen, L.; Chen, J.; Shen, X.S. Diverse grouping-based aggregation protocol with error detection for smart grid communications. *IEEE Trans. Smart Grid* **2015**, *6*, 2856–2868. [[CrossRef](#)]
14. Gupta, S.S.; Maitra, S.; Paul, G.; Sarkar, S. (Non-)Random Sequences from (Non-)Random Permutations—Analysis of RC4 Stream Cipher. *J. Cryptol.* **2014**, *27*, 67–108. [[CrossRef](#)]
15. Mahmood, K.; Chaudhry, S.A.; Naqvi, H.; Shon, T.; Ahmad, H.F. A lightweight message authentication scheme for smart grid communications in power sector. *Comput. Electr. Eng.* **2016**, *52*, 114–124. [[CrossRef](#)]
16. Li, H.; Lu, R.; Zhou, L.; Yang, B.; Shen, X. An Efficient Merkle-Tree-Based Authentication Scheme for Smart Grid. *IEEE Syst. J.* **2014**, *8*, 655–663. [[CrossRef](#)]
17. Chim, T.W.; Yiu, S.M.; Li, V.O.K. PRGA: Privacy-Preserving Recording Gateway-Assisted Authentication of Power Usage Information for Smart Grid. *IEEE Trans. Depend. Secur. Comput.* **2015**, *12*, 85–97. [[CrossRef](#)]
18. Fan, C.I.; Huang, S.Y.; Lai, Y.L. Privacy-Enhanced Data Aggregation Scheme Against Internal Attackers in Smart Grid. *IEEE Trans. Ind. Inform.* **2013**, *10*, 666–675. [[CrossRef](#)]
19. Castelluccia, C.; Mykletun, E.; Tsudik, G. Efficient Aggregation of encrypted data in Wireless Sensor Networks. In Proceedings of the 2th International Conference on Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS'05), San Diego, CA, USA, 17–21 July 2005; pp. 109–117.
20. Dimitriou, T.; Awad, M.K. Secure and scalable aggregation in the smart grid resilient against malicious entities. *Ad Hoc Netw.* **2016**, *50*, 58–67. [[CrossRef](#)]
21. Rahman, M.A.; Manshaei, M.H.; Al-Shaer, E.; Shehab, M. Secure and Private Data Aggregation for Energy Consumption Scheduling in Smart Grids. *IEEE Trans. Depend. Secur. Comput.* **2017**, *14*, 221–234. [[CrossRef](#)]
22. Shamir, A. How to Share a Secret. *Commun. ACM* **1979**, *22*, 612–613. [[CrossRef](#)]
23. Erkin, Z.; Tsudik, G. Private computation of spatial and temporal power consumption with smart meters. In Proceedings of the 10th International Conference on Applied Cryptography and Network Security (ACNS'12), Singapore, 26–29 June 2012; pp. 561–577.
24. Jia, W.; Zhu, H.; Cao, Z.; Dong, X.; Xiao, C. Human-factor-aware privacy-preserving aggregation in smart grid. *IEEE Syst. J.* **2017**, *18*, 598–607. [[CrossRef](#)]
25. FriendlyARM. 2011. Available online: <http://www.friendlyarm.net/> (accessed on 17 August 2011).
26. Ahlswede, R.; Csiszar, I. Common randomness in information theory and cryptography I. Secret sharing. *IEEE Trans. Inform. Theory* **1993**, *39*, 1121–1132. [[CrossRef](#)]