IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 8, 2021, accepted May 4, 2021, date of publication May 10, 2021, date of current version May 18, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3078629

Efficient and Privacy-Preserving Massive Data Processing for Smart Grids

HUA SHEN[1,2], MINGWU ZHANG[1,2], HAO WANG[3], FUCHUN GUO[4], AND WILLY SUSILO[4], (Fellow, IEEE)

[1]School of Computer Science, Hubei University of Technology, Wuhan 430068, China
[2]Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China
[3]School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China
[4]School of Computing and Information Technology, Institute of Cybersecurity and Cryptology, University of Wollongong, Wollongong, NSW 2522, Australia

Corresponding author: Mingwu Zhang (csmwzhang@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702168, Grant 62072134, and Grant U2001205; in part by the Guangxi Key Laboratory of Trusted Software under Grant kx202014, in part by the Natural Science Foundation of Guangxi Province under Grant 2019JJD170020, and in part by the State Key Laboratory of Information Security of China under Grant 2020-MS-05.

ABSTRACT Analysis and utilization of massive meter data can help decision-makers provide reasonable decisions. Therefore, multi-functional meter data processing has received considerable attention in recent years. Nevertheless, it might compromise users' privacy, such as releasing users' lifestyles and habits. In this paper, we propose an efficient and privacy-preserving massive data process for smart grids. The presented protocol utilizes the Paillier homomorphic encryption and Horner's Rule to achieve a privacy-preserving two-level random permutation method, making large-scale meter data permuted randomly and sufficiently in a privacy-preserving way. As a result, the analysis center can simultaneously implement various data processing functions (such as variance, comparing, linear regression analysis), and it does not know the source of data. The security analysis shows that our protocol can realize data confidentiality and data source anonymity. The detailed analyses demonstrate that our protocol is efficient in terms of computational and communication costs. Furthermore, it can support fault tolerance of entity failures and has flexible system scalability.

INDEX TERMS Privacy preservation, efficient data processing, random permutation, massive meter data, smart grids.

I. INTRODUCTION

Smart grids add communication networks to the traditional electrical grid infrastructure [1]. In smart grids, massive smart meters [2]–[4] are deployed to collect near-real-time users' meter data, making the electrical grid more reliable and efficient [1]. Analyzing and utilizing massive meter data can help policy-makers provide reasonable decisions and aid people in improving their quality of life. For example, the analysis of meter data can help power companies develop more reasonable sub-tariff policies, more accurately predicting users' electricity consumption, and better real-timely controlling the power resources. It can also aid users in utilizing electricity rationally and avoiding the peak. Nevertheless, meter data are directly related to the state of users' activities and home appliances, which pose significant users' privacy risks [5], [6]. For instance, if one user's meter data is low from 9 a.m. to 11 a.m., that indicates the user is not at home with a high probability. Loss of control over data can lead to data leakage [7], [8]. Therefore, encrypting the meter data before being uploaded to the appropriate organization for processing is a good way [7], [9]. The homomorphic encryption can convert operations on plaintexts into operations on ciphertexts [10], [11] and is a powerful tool to perform privacy-preserving data processing [12]. Literature [11], [13]–[19] usually adopts semi-homomorphic encryption (such as the Paillier encryption, the ElGamal encryption, the Lifted ElGamal encryption) or somewhat homomorphic encryption (such as BGN) to realize privacy-preserving data processing. Nevertheless, some limitations are: 1) The raw data need to be encrypted, and some algebraic structures about the raw data also need to be encrypted. Besides, different data processing functions require different algebraic structures to be encrypted. For example, for computing the variance of

The associate editor coordinating the review of this manuscript and approving it for publication was Amit Singh.

data, the raw data and their square both need to be encrypted; for calculating data's Euclidean distance, the raw data, their negative form, and their square need to be encrypted. Usually, the data processed requirements in one practical application, even over the same data set, are diverse (such as summation, average, variance, min/max, linear regression analysis). Therefore, multifunctional data processing by using homomorphic encryption is a challenge. 2) The homomorphic encryption operations bring heavy computational and communication burdens. Therefore, massive data processing by using homomorphic encryption is also a challenge. To sum up, the efficient processing of massive meter data in a privacy-preserving manner for smart grids is still an enormous challenge. We present an efficient and privacy-preserving massive data processing (PPMDP) protocol for smart grids for this challenge. Note that efficiency here has two meanings: one is the high efficiency of processing massive meter data, and the other is the wide range of data processing functions that can be implemented simultaneously.

For massive meter data processing, we note that the fog paradigm is well-positioned for large-scale data analysis [20]–[24]. Fog devices, which can be gateways or aggregators nearby neighborhoods, are at the network edge to extend the computing and data processing capabilities to the network edge. Therefore, we introduce the fog paradigm into our system model to assist in handling massive meter data. Based on our system model, we utilize homomorphic encryption to realize the random permutation of massive meter data to destroy the linking relationship between users and their meter data, ensuring our protocol PPMDP can protect data privacy. Because finally, the data analysis center processes original meter data, PPMDP can achieve plenty of data processing functions, making better computational and communication efficiency. In a nutshell, our specific contributions are threefold.

(1) To realize plenty of privacy-preserving data processing functions simultaneously, we present a privacy-preserving two-level random permutation method to adequately and securely break the links between massive meter data and their sources. It makes the analysis center process raw meter data but does not know whose data they are. We utilize the Paillier homomorphic encryption and Horner's Rule to achieve this method.

(2) To process large-scale meter data in one-round communication, we present a three-layer system architecture. The first layer is composed of massive users, who provide massive meter data. The second layer consisting of fog devices and cloud servers executes the two-level random permutation. The top layer includes a data analysis center, which performs plenty of functions over massive meter data. Also, there is a logical hierarchy to organize enormous users. We logically divide users into many groups, and each group has at most n users. Then, we divide groups into a few clusters, and each cluster has at most m groups. In other words, each cluster has at most mn users. One fog device is in charge of one group, and one cloud server corresponds with one cluster. In our system model, there are η such clusters. That is, the amount of data handled by our system is at most ηmn.

(3) Our protocol PPMDP can implement secure multifunctional calculations of massive meter data, support fault tolerance of entities' failures, and has better system scalability. The performance analysis demonstrates that it is efficient in terms of computational costs and communication overhead.

The rest of this paper is organized as follows. We discuss the related works in section II. In section III, we describe our system model, threat model, design goals. In section IV, we review some preliminaries. In section V, we elaborate on the construction of our PPMDP protocol, followed by correctness discussion, security analysis, feature analysis, and performance analysis in sections VI, VII, VIII, and IX, respectively. Finally, we draw our conclusion in section X.

II. RELATED WORK

For implementing privacy-preserving data processing, some works [2], [25], [26] use differential privacy technology, and some works [1], [27], [28] adopt secure multiparty computation technology. However, these works either reduce data accuracy and affect data availability or require multiple communication and cooperation rounds between data holders. High data availability and one round of communication and cooperation between entities are our design goals. Therefore, we focus on homomorphic encryption technologies.

Several privacy-preserving data handling protocols by utilizing homomorphic encryption technologies [13], [14], [16], [17] have been proposed in the last decade. We observe that the protocols adopting semi-homomorphic encryption need to construct encrypted data items according to objective functions. For example, according to the target average and variance functions, Lu *et al.* [14] constructed encrypted data items using the Chinese Remainder Theorem. One such data item is a kind of algebraic structure of one device's data and its square. If the target function is to obtain summation, Liu *et al.* [13] leveraged the Lifted EC-ElGamal cryptosystem to encrypt each user's meter data. Mustafa *et al.* [29] and Mahdikhani *et al.* [30] utilized the Paillier encryption to encrypt users' electricity consumption data and attribute value, respectively. To gain Euclidean distances [12], the x-coordinate and y-coordinate of one position and their squares are needed to be encrypted. Therefore, adopting half homomorphic encryption to implement data processing is not very convenient. The BGN cryptosystem has many-times additive and one-time multiplicative homomorphism properties. As a result, the protocols employing the BGN cryptosystem usually can realize some functions on the ciphertexts of original data at the same time. For instance, Chen *et al.* [16] used the BGN cryptosystem to securely calculate data's summation, average, variance, and F-test. Han *et al.* [17] and Ren *et al.* [18] leveraged the BGN cryptosystem to implement the secure computation of data's summation, average,

min/max, and some other functions, respectively. However, the BGN cryptosystem only supports one time multiplicatively homomorphic operation. Therefore, using the BGN cryptosystem, these schemes can merely calculate the functions with the highest exponent order of 2 securely.

Some privacy-preserving data handling protocols based on data anonymity have been presented. [6] hides the link between data and its source using virtual groups' private keys to encrypt users' data. However, this protocol has a complicated key generation and management process. In [31], Zhang *et al.* first propose a bitwise XOR homomorphic cipher system and subsequently present an efficient protocol to achieve n-source anonymity (suppose there are n users) in mobile phone sensing, which can realize the secure computation of arbitrary/complex functions. Whereas, there are two main limitations in the work of [31]. First, each user needs to report an encrypted ln-bit string. Only l-bit substring is the ciphertext of the original data, other $l(n-1)$-bit substring is the ciphertext of $n-1$ dummy data, which results in the communication overhead is not satisfactory. Second, the position of every user's original data in the encrypted ln-bit string is fixed, which possibly reduces the security of this protocol.

III. PROBLEM FORMALIZATION

This section gives the system model, the threat model, the design goals, and our fundamental idea of secure data perturbing used in our protocol. For ease of reading, Table 1 describes the main notations in this paper.

TABLE 1. Main notations.

Notation	Description
p, q, g, λ, μ, N	Parameters of the Paillier Encryption
R_1, R_2	Parameters of Horner's Rule
AC, CS	Analysis center, Cloud server
m	Number of FDs
FD_i	ith Fog device ($i = 1, 2, \cdots, m$)
n_i	Number of Users within the jurisdiction of a FD_i
n	$n \geq max\{n_1, n_2, \cdots, n_m\}$
User_{ij}	jth User of FD_i ($j = 1, 2, \cdots, n_i$)
d_{ij}	Meter data of User_{ij}
δ	d_{ij} can be expressed as a δ-digit ternary number
$d_{ij}^{(k)}$	kth ternary bit of d_{ij} ($k = 1, 2, \cdots, \delta$)
$C_{ij}^{(k)}$	the ciphertext of $d_{ij}^{(k)}$
C_{ij}	$C_{ij} = C_{ij}^{(\delta-1)} \| \cdots \| C_{ij}^{(0)}$
$\{p_{i1}, p_{i2}, \cdots, p_{in_i}\}$	A random arrangement of $\{1, 2, \cdots, n_i\}$ generated by FD_i
$C_{ij}^{\prime(k)}$	the permuted $C_{ij}^{(k)}$ after a random permutation operation according to $\{p_{i1}, p_{i2}, \cdots, p_{in_i}\}$
$C_{ij}^{\prime(k)}$	the permuted $C_{ij}^{(k)}$ after a random permutation operation according to $\{p_{i1}, p_{i2}, \cdots, p_{in_i}\}$
$\bar{C}_i^{(k)}$	the aggregation ciphertext of $C_{i1}^{\prime(k)}$, $C_{i2}^{\prime(k)}$, $\cdots$, $C_{in_i}^{\prime(k)}$
$\bar{C}_i$	$\bar{C}_i = \bar{C}_i^{(\delta-1)} \| \cdots \| \bar{C}_i^{(0)}$
$\{p_1, p_2, \cdots, p_m\}$	A random arrangement of $\{1, 2, \cdots, m\}$ generated by CS
$\bar{C}_i^{\prime(k)}$	the permuted $\bar{C}_i^{(k)}$ after a random permutation operation according to $\{p_1, p_2, \cdots, p_m\}$
$\tilde{C}^{(k)}$	the aggregation ciphertext of $\bar{C}_1^{\prime(k)}$, $\bar{C}_2^{\prime(k)}$, $\cdots$, $\bar{C}_m^{\prime(k)}$
$\tilde{C}$	$\tilde{C} = \tilde{C}^{(\delta-1)} \| \cdots \| \tilde{C}^{(0)}$

A. SYSTEM MODEL

Our system model (shown in Fig.1) includes three layers: Data Providing Layer, Data Perturbing Layer, and Data Processing Layer. Large-scale users are in the Data Providing Layer and are logically divided into many groups. Moreover, these groups are organized into several clusters. The Data Perturbing Layer consists of fog devices and cloud servers. One fog device is in charge of one group, and one cloud server is responsible for one cluster. There are at most m groups in each cluster, and most n users in each group. In other words, each cluster can cover at most mn users. If there are η clusters in our system, the total number of users is ηmn. **Note that** the system model in Fig.1 is a kind of logical system structure. In actual deployment, a cloud server may be in charge of several clusters, and a fog device probably manages several groups. In what it follows, we describe the four entities involved in our system model in detail: User, Fog Devices (FD), Cloud Server (CS), and Analysis Center (AC).

User: Users are responsible for collecting and providing the meter data.

Fog Device, FD: FD realizes the random permutation of a group in a privacy-preserving manner.

Cloud Server, CS: CS accomplishes the random permutation of a cluster in a privacy-preserving manner.

Analysis Center, AC: AC flexibly handles and analyzes raw meter data and does not know the source of each data.

B. THREAT MODEL

The following attacker models are assumed:

(1) AC, CS, and FD are honest-but-curious. They keep the system running smoothly without launching an active attack. However, they keep all inputs, intermediate results and try to infer meter data.

(2) Users are honest-but-curious. They do not maliciously discard or distort any raw meter data and follow the protocol correctly. However, they seek to obtain as much information about other users' meter data.

(3) CS, FD, and several users may collude to discover other users' meter data. It must be noted that the collusion of AC and other entities is out of scope.

(4) $\mathscr{A}$ is an external adversary. $\mathscr{A}$ tries to gain the privacy information of any meter data through eavesdropping on communication channels and intruding in the databases of FD, CS, and AC. Note that $\mathscr{A}$ cannot obtain the secret parameters of entities through intruding in their databases.

In this paper, for AC, we view data sources as data privacy. For other entities, we regard data content as data privacy. Specifically, in our protocol, we focus on meter data confidentiality and anonymity.

Definition 1: (Data Source Anonymity). Suppose the possible set of users be U. Let $|U| = n$. The source anonymity of a meter data d is satisfied if no adversary can find the source of d with a probability higher than $\frac{1}{n}$ when it obtains d. In nature, data source anonymity here is k-anonymity [32].

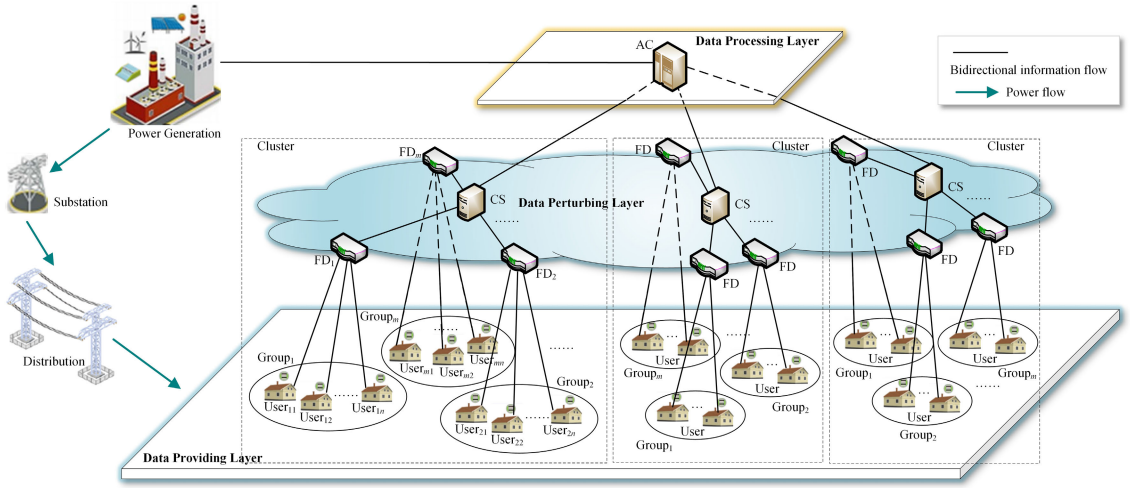

FIGURE 1. System model of our protocol PPMDP.

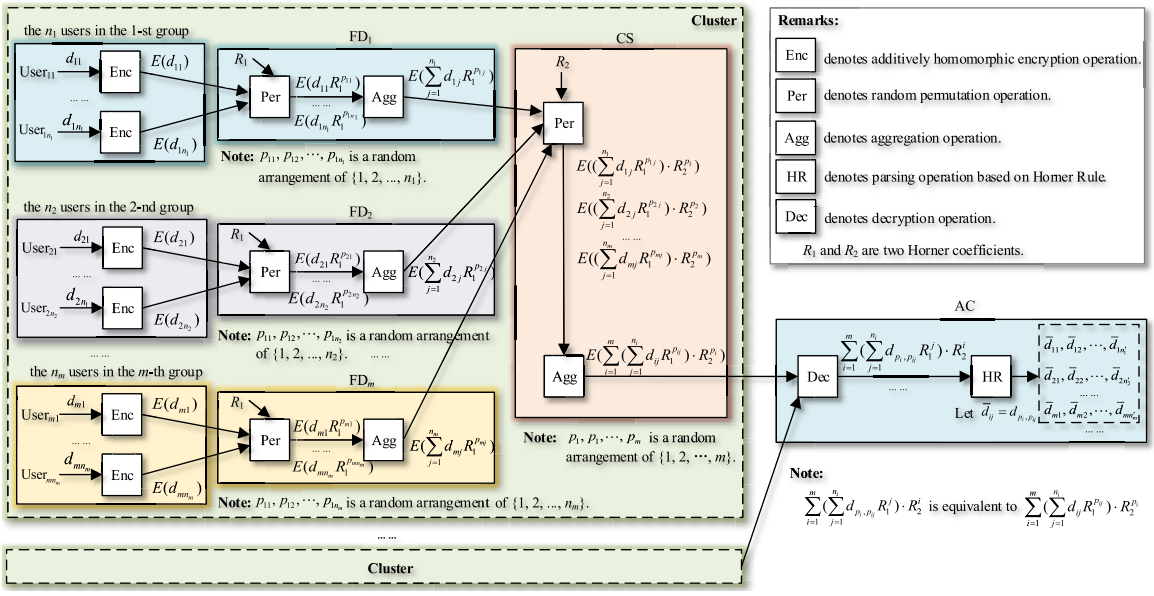

FIGURE 2. Basic idea of our protocol PPMDP.

Note that $\mathscr{A}$ may launch other active attacks on data integrity and source authentication. However, since we focus on meter data confidentiality and anonymity in this paper, those active attacks are beyond the scope of this paper, although it is not difficult to apply some sophisticated digital signature techniques to tackle these attacks.

C. DESIGN GOAL

The main goal of PPMDP is to provide efficient multi-functional processing for large-scale meter data with guaranteeing data confidentiality and data source anonymity. Specifically, the goals of PPMDP are:

Data Confidentiality: The confidentiality of meter data has to be guaranteed, such that AC is the only entity that obtains raw meter data.

Data Source Anonymity: The sources of meter data (that is, users) remain unconditionally anonymous to AC.

Data Processing Diversity: PPMDP can simultaneously realize many functions in a privacy-preserving manner, which is more in line with the application requirements of efficiently handling massive meter data.

Efficiency: PPMDP can reduce transmission load and computational costs as much as possible.

D. BASIC IDEA

Our basic idea is shown as Fig.2. We take one cluster as an example. d_{ij} is the meter data of the jth user of the ith group (i.e., User_{ij}), where $i = 1, 2, \cdots, m$, $j = 1, 2, \cdots, n_i$. User_{ij} encrypts d_{ij} and sends it to FD_i. FD_i performs a random permutation operation on its received ciphertexts, and then

aggregates them and transmits the result to CS. That is, CS obtains $Enc(\sum_{j=1}^{n_i} d_{ij} \cdot R_1^{p_{ij}})$ (simplified as $Enc(poly_i)$), where $\{p_{i1}, p_{i2}, \cdots, p_{in_i}\}$ is a random arrangement of $\{1, 2, \cdots, n_i\}$. The coefficient of $poly_i$'s jth item is d_{ij} with a probability of $\frac{1}{n_i}$. Similarly, CS first executes a random permutation operation on its received ciphertexts, and then aggregates them and sends the result to AC. AC attains $Enc(\sum_{i=1}^{m} poly_i \cdot R_2^{p_i})$ (simplified as $Enc(poly)$). The coefficient of $poly$'s ith item is $poly_i$ with a probability of $\frac{1}{m}$. Through decrypting $Enc(poly)$ and parsing $poly$, AC will obtain all meter data without knowing their source. Note that the process is the same for each cluster. If there are η clusters, AC eventually receives η ciphertexts.

IV. PRELIMINARIES

Here, we briefly recall the Paillier cryptosystem [12], [33] and Horner's Rule [11], which are used throughout this work.

A. PAILLIER CRYPTOSYSTEM

The Paillier cryptosystem consists of three algorithms:

Key Generation: Let $N = pq$ and p, q are two κ-bit prime numbers, and κ is the security parameter. Let $\lambda = lcm(p-1, q-1)$, define a function $L(\mu) = (\mu - 1)/N$, pick a random generator $g \in \mathbb{Z}_{N^2}^*$, and calculate $\mu = (L(g^\lambda \ mod \ N^2)) - 1$. The public key is (N, g), and the private key is (λ, μ).

Encryption: To encrypt a message $m \in \mathbb{Z}_N$, we choose a random number $r \in \mathbb{Z}_N^*$, and calculate the ciphertext $C = g^m \cdot r^N \mod N^2$.

Decryption: Given the ciphertext C, the corresponding message can be recovered by computing $m = L(C^\lambda \bmod N^2) \cdot \mu$.

The Paillier cryptosystem has the following homomorphic properties: $E(m_1, r_1)E(m_2, r_2) = E(m_1 + m_2, r_1 \cdot r_2)$ and $E(m_1, r_1)^{m_2} = E(m_1 \cdot m_2, r_1^{m_2})$.

B. HORNER'S RULE

We can use Horner's Rule to redescribe any polynomial $p(R) = a_n R^n + a_{n-1}R^{n-1} + \cdots + a_1 R$ as $p(R) = (\cdots(a_n R + a_{n-1})R + \cdots) + a_1)R$. If we know $p(R)$ and R, where $R > max\{a_n, a_{n-1}, \cdots, a_1\}$, we can obtain the coefficients $a_1, a_2, \cdots, a_n$ of the polynomial $p(R)$ through n divisible operations and n modulo operations. The limitation of Horner's Rule is $p(R)$ grows exponentially with the parameter R. Therefore, we should choose the parameter R as little as possible.

V. OUR PROPOSED PROTOCOL

In this section, we describe our protocol PPMDP, which consists of five phases: System Initialization, Data Encryption, Local Perturbing, Global Perturbing, and Data Processing. We take one cluster as an example to expound our protocol PPMDP. Note that the process procedure is the same for each cluster, and all clusters execute PPMDP in parallel. In our system model, one cluster has at most m groups, and one group has at most n users. We denote the number of users in the ith group as n_i ($n_i \le n$). To simplify the description, we assume one cluster includes m groups. User_{ij} is the jth user of ith group, where $1 \le i \le m$, $1 \le j \le n_i$. And we denote User_{ij}'s meter data as d_{ij}.

To overcome the limitation of Horner's Rule, we should choose a little Horner parameter R as possible. The minimum value of R is 2. At this moment, meter data should be expressed as a binary number. Each user generates a ciphertext for each binary bit of his meter data. Tradeoff the number of ciphertexts generated by each user and the number of users of each cluster, in our protocol, we express a meter data as a δ-digit ternary number. So the upper bond W of a meter data is $2 * 3^{\delta-1} + 2 * 3^{\delta-2} + \cdots + 2 * 3^1 + 2 * 3^0$.

A. SYSTEM INITIALIZATION

In this part, given a security parameter κ, the system generates AC's public and private key pair ($pk = (N, g)$, $sk = (\lambda, \mu)$) by calling the Key Generation algorithm of the Paillier encryption. And then, the system chooses two public parameters (that is, two Horner parameters) $R_1 = 3$ and $R_2 = 3^{n+1}$, where $n \ge max\{n_1, n_2, \cdots, n_m\}$, $R_2 < \sqrt[m+1]{N/2}$.

B. DATA ENCRYPTION

User_{ij} collects its meter data d_{ij} at time point t, and then executes the following steps.

Step 1.1: User_{ij} transforms d_{ij} to the corresponding ternary number $(d_{ij}^{(\delta-1)}, d_{ij}^{(\delta-2)}, \cdots, d_{ij}^{(1)}, d_{ij}^{(0)})$. Then, for all $k = 0, 1, \cdots, \delta - 1$, User_{ij} chooses a random number $r_{ij}^{(k)} \in \mathbb{Z}_N^*$ and generates the ciphertext as

$$C_{ij}^{(k)} = g^{d_{ij}^{(k)}} \cdot (r_{ij}^{(k)})^N \bmod N^2 \tag{1}$$

Note that User_{ij} executes Step 1.1 δ times (that is, $k = 0, 1, \cdots, \delta - 1$), and finally obtains $C_{ij}^{(0)}, C_{ij}^{(1)}, \cdots, C_{ij}^{(\delta-1)}$.

Step 1.2: User_{ij} delivers $C_{ij} = C_{ij}^{(\delta-1)} \,||\, \cdots \,||\, C_{ij}^{(0)}$ to FD_i.

C. LOCAL PERTURBING

Upon receiving the ciphertexts from n_i users, FD_i performs the following steps.

Step 2.1: FD_i randomly picks $p_{ij} \in \{1, 2, \cdots, n_i\} \setminus \{p_{i1}, p_{i2}, \cdots, p_{i(j-1)}\}$ and calculates

$$\begin{aligned} C'^{(k)}_{ij} &= (C_{ij}^{(k)})^{R_1^{p_{ij}}} \\ &= g^{R_1^{p_{ij}} \cdot d_{ij}^{(k)}} \cdot ((r_{ij}^{(k)})^{R_1^{p_{ij}}})^N \bmod N^2 \end{aligned} \tag{2}$$

Step 2.2: FD_i aggregates $(C'^{(k)}_{i1}, C'^{(k)}_{i2}, \cdots, C'^{(k)}_{in_i})$ to obtain

$$\begin{aligned} \bar{C}_i^{(k)} &= \prod_{j=1}^{n_i} C'^{(k)}_{ij} \\ &= g^{\sum_{j=1}^{n_i} R_1^{p_{ij}} \cdot d_{ij}^{(k)}} \cdot \prod_{j=1}^{n_i} ((r_{ij}^{(k)})^{R_1^{p_{ij}}})^N \bmod N^2 \end{aligned} \tag{3}$$

Note that FD_i executes Steps 2.1 and 2.2 δ times (that is, $k = 0, 1, \cdots, \delta - 1$), and finally obtains $\bar{C}_i^{(0)}, \bar{C}_i^{(1)}, \cdots, \bar{C}_i^{(\delta-1)}$.

Step 2.3: FD_i sends $\bar{C}_i = \bar{C}_i^{(\delta-1)} \,||\, \cdots \,||\, \bar{C}_i^{(0)}$ to CS.

D. GLOBAL PERTURBING

After receiving the aggregation ciphertexts from m FDs, CS completes the following steps.

Step 3.1: CS randomly picks $p_i \in \{1, 2, \cdots, m\} \setminus \{p_1, p_2, \cdots, p_{i-1}\}$ and calculates

$$\begin{aligned}\bar{C'}_i^{(k)} &= (\bar{C}_i^{(k)})^{R_2^{p_i}} \\ &= (g^{R_2^{p_i}(\sum_{j=1}^{n_i} R_1^{p_{ij}} \cdot d_{ij}^{(k)})} \cdot \prod_{j=1}^{n_i}((r_{ij}^{(k)})^{R_2^{p_i} \cdot R_1^{p_{ij}}})^N) \bmod N^2 \end{aligned} \quad (4)$$

Step 3.2: CS aggregates $(\bar{C'}_1^{(k)}, \bar{C'}_2^{(k)}, \cdots, \bar{C'}_m^{(k)})$ to obtain

$$\begin{aligned}\tilde{C}^{(k)} &= \prod_{i=1}^{m} \bar{C'}_i^{(k)} \\ &= g^{\sum_{i=1}^{m} R_2^{p_i} \cdot (\sum_{j=1}^{n_i} R_1^{p_{ij}} \cdot d_{ij}^{(k)})} \\ &\quad \cdot \prod_{i=1}^{m}\prod_{j=1}^{n_i}((r_{ij}^{(k)})^{R_2^{p_i} \cdot R_1^{p_{ij}}})^N \bmod N^2 \end{aligned} \quad (5)$$

Note that CS executes Steps 3.1 and 3.2 δ times (that is, $k = 0, 1, \cdots, \delta - 1$), and finally obtains $\tilde{C}_i^{(0)}, \tilde{C}_i^{(1)}, \cdots, \tilde{C}_i^{(\delta-1)}$.

Step 3.3: CS sends $\tilde{C} = \tilde{C}^{(\delta-1)} \,||\, \cdots \,||\, \tilde{C}^{(0)}$ to AC.

E. DATA PROCESSING

AC can recover $M^{(k)}$ by executing the Paillier decryption algorithm.

$$\begin{aligned}\tilde{C}^{(k)} &= g^{\sum_{i=1}^{m} R_2^{p_i} \cdot (\sum_{j=1}^{n_i} R_1^{p_{ij}} \cdot d_{ij}^{(k)})} \\ &\quad \cdot \prod_{i=1}^{m}\prod_{j=1}^{n_i}((r_{ij}^{(k)})^{R_2^{p_i} \cdot R_1^{p_{ij}}})^N \bmod N^2 \\ &= g^{M^{(k)}} \cdot (R^{(k)})^N \bmod N^2 \end{aligned} \quad (6)$$

where

$$\begin{aligned}M^{(k)} &= \sum_{i=1}^{m} R_2^{p_i} \cdot (\sum_{j=1}^{n_i} R_1^{p_{ij}} \cdot d_{ij}^{(k)}) \\ &= \sum_{i=1}^{m} R_2^{i} \cdot (\sum_{j=1}^{n_{p_i}} R_1^{j} \cdot d_{p_i p_{ij}}^{(k)}) = \sum_{i=1}^{m} R_2^{i} \cdot (\sum_{j=1}^{n'_i} R_1^{j} \cdot \bar{d}_{ij}^{(k)}) \end{aligned} \quad (7)$$

$$\begin{aligned}R^{(k)} &= \prod_{i=1}^{m}\prod_{j=1}^{n_i}(r_{ij}^{(k)})^{R_2^{p_i} \cdot R_1^{p_{ij}}} \\ &= \prod_{i=1}^{m}\prod_{j=1}^{n_{p_i}}(r_{p_i p_{ij}}^{(k)})^{R_2^{i} \cdot R_1^{j}} = \prod_{i=1}^{m}\prod_{j=1}^{n'_i}(\bar{r}_{ij}^{(k)})^{R_2^{i} \cdot R_1^{j}} \end{aligned} \quad (8)$$

where n_{p_i} is denoted as n'_i. And then, AC executes Algorithm 1 with $M^{(k)}, R_1, R_2, m, n'_1, n'_2, \cdots, n'_m$ as inputs, and obtains $(\bar{d}_{11}^{(k)}, \bar{d}_{12}^{(k)}, \cdots, \bar{d}_{1n'_1}^{(k)}), \cdots, (\bar{d}_{m1}^{(k)}, \bar{d}_{m2}^{(k)}, \cdots, \bar{d}_{mn'_m}^{(k)})$. After converting these results to the corresponding decimal number, AC can achieve efficient computation of many functions (such as variance, comparing, linear regression analysis) on raw meter data. AC needs to execute δ times Algorithm 1, the total time complexity is $O(\delta m \cdot max\{n_1, n_2, \cdots, n_m\})$. Note that the work of this paper does not apply to such analyses that require positioning or tracking (for instance, finding out the users who use tremendous electricity), because of breaking the correspondences between data and their sources.

Algorithm 1 Parsing Processing With Horner's Rule

Input: $M, R_1, R_2, m, n'_1, n'_2, \cdots, n'_m$
Output: $(a_{11}, a_{12}, \cdots, a_{1n'_1}), \cdots, (a_{m1}, a_{m2}, \cdots, a_{mn'_m})$

```
X_0 = M/R_2;
for i = 1 to m do
    M_i = X_{i-1} mod R_2; X_i = X_{i-1}/R_2;
end for
for i = 1 to m do
    X_{i0} = M_i/R_1;
    for j = 1 to n'_i do
        a_ij = X_{i(j-1)} mod R_1; X_ij = X_{i(j-1)}/R_1;
    end for
end for
return (a_11, a_12, ..., a_1n'_1), ..., (a_m1, a_m2, ..., a_mn'_m);
```

VI. CORRECTNESS DISCUSSION

Because for all $k = 0, 1, \cdots, \delta - 1$, $i = 1, 2, \cdots, m$, and $j = 1, 2, \cdots, n_i$, we have $d_{ij}^{(k)} \leq 2$, so we set $R_1 = 3$, $R_2 = 3^{n+1}$, and ensure $R_2 < \sqrt[m+1]{N/2}$. Let $n \geq max\{n_1, n_2, \cdots, n_m\}$.

Because of $\sum_{j=1}^{n_i} R_1^j \cdot d_{ij}^{(k)} \leq \sum_{j=1}^{n} 3^j \cdot 2 = 3^{n+1} - 3 < 3^{n+1} = R_2$, and $d_{ij}^{(k)} \leq 2 < 3 = R_1$, which satisfies the requirement of the parameter of Horner's Rule (see Subsection IV-B), so we can obtain correct $d_{ij}^{(k)}$ by executing Alg.1 with taking $\sum_{j=1}^{n_i} R_1^j \cdot d_{ij}^{(k)}$, $R_1 = 3$, and $R_2 = 3^{n+1}$ as inputs.

According to the Paillier cryptosystem, the correct decryption of the ciphertext $\tilde{C}^{(k)}$ must satisfy the condition $M^{(k)} < N$, i.e., $\sum_{i=1}^{m} R_2^i \cdot (\sum_{j=1}^{n'_i} R_1^j \cdot \bar{d}_{ij}^{(k)}) < N$ (see (6), (7)). We have

$$\begin{aligned}\sum_{i=1}^{m} R_2^i \cdot (\sum_{j=1}^{n'_i} R_1^j \cdot \bar{d}_{ij}^{(k)}) &< 2 \cdot \sum_{i=1}^{m} R_2^i \cdot (\sum_{j=1}^{n'_i} R_1^j) \\ &< 2 \cdot \sum_{i=1}^{m} R_2^i \cdot (\sum_{j=1}^{n} 3^j) < 3^{n+1} \cdot \sum_{i=1}^{m} R_2^i \\ &= R_2 \cdot \frac{R_2^{m+1} - R_2}{R_2 - 1} < \frac{R_2}{R_2 - 1} \cdot R_2^{m+1} \\ &< 2 \cdot R_2^{m+1} < 2 \cdot (\sqrt[m+1]{N/2})^{m+1} = N \end{aligned} \quad (9)$$

Therefore, AC can correctly decrypt $\tilde{C}^{(k)}$ to obtain $M^{(k)}$.

VII. SECURITY ANALYSIS

Theorem 1: Data confidentiality can be achieved in the proposed protocol.

Proof: In PPMDP, before delivering, users encrypt their meter data by executing the Paillier encryption algorithm

with AC's public key. Moreover, FDs and CS realize the local perturbing (Subsection V-C) and the global perturbing (Subsection V-D) in ciphertext space, respectively. Based on the parameter setting discussed in section VI, C_{ij}, $C'^{(k)}_{ij}$, $\bar{C}^{(k)}_i$, $\bar{C}'^{(k)}_i$, and $\tilde{C}^{(k)}$ are valid ciphertexts. Since the Paillier cryptosystem is semantic secure and AC's private key is secretly protected, neither FDs and CS nor $\mathscr{A}$ can obtain any useful information from these ciphertexts, even in the case of collusion of some users, multiple FDs, and CS. Therefore, PPMDP can guarantee the confidentiality of meter data.

Theorem 2: Data source anonymity can be guaranteed in the proposed protocol.

Proof: In PPMDP, based on the parameter R_1, FD_i stores n_i meter data into a polynomial (denoted as $poly_i$) by executing the ciphertext operations (2) and (3). The polynomial $poly_i$ satisfies Horner's Rule. In $poly_i$, the jth coefficient is the jth user's meter data with $\frac{1}{n_i}$ probability. Similarly, based on the parameter R_2, CS embeds $poly_i$ into a polynomial (denoted as $poly$) by carrying out the ciphertext operations (4) and (5). The polynomial $poly$ satisfies Horner's Rule. The ith coefficient of $poly$ is $poly_i$ with $\frac{1}{m}$ probability. By decrypting and parsing the ciphertext from CS, AC obtains $\bar{d}_{ij}$. The probability that $\bar{d}_{ij}$ is the meter data of User_{ij} is $\frac{1}{m \cdot n_i}$. Hence, AC's probability of successfully guessing the source of $\bar{d}_{ij}$ is $\frac{1}{m \cdot n_i}$, which is lower than $\frac{1}{n_i}$. Moreover, even if $\mathscr{A}$ intrudes into AC's database to obtain $\bar{d}_{ij}$ and colludes with CS, it maybe knows $\bar{d}_{ij}$ comes from which group, but its probability of successfully guessing the source of $\bar{d}_{ij}$ is $\frac{1}{n_i}$, which is not higher than $\frac{1}{n_i}$. Therefore, PPMDP can guarantee the anonymity of meter data.

VIII. FEATURE ANALYSIS

In this section, we analyze the characters of PPMDP in terms of fault tolerance and system scalability.

A. FAULT TOLERANCE

PPMDP can still work well even when some entities failure and can support fault tolerance of user failures, fog device failures, and cloud server failures. If some users malfunction, the corresponding fog device (suppose it is FD_i) cannot receive the reports of these users. FD_i counts the number of reports received (assume the number is $\bar{n}_i$) and then disturbs and aggregates these reports. It randomly picks $p_{ij} \in \{1, \cdots, \bar{n}_i\} \setminus \{p_{i1}, \cdots, p_{i(j-1)}\}$ and calculates the (2). And then, It aggregates $(C'_{i1}, \cdots, C'_{i\bar{n}_i})$ to obtain $\hat{C}_i = \prod_{j=1}^{\bar{n}_i} C'_{ij}$. Subsequently, FD_i transmits the result to CS. If some FDs do not work, CS cannot receive the reports of these FDs. CS counts the number of its received reports. After permutating these reports randomly, CS aggregates them. CS randomly picks $p_i \in \{1, \cdots, \bar{m}\} \setminus \{p_1, \cdots, p_{i-1}\}$ and calculates the (4). And then, CS aggregates $(\bar{C}'_1, \cdots, \bar{C}'_{\bar{m}})$ to obtain $\tilde{C} = \prod_{i=1}^{\bar{m}} \bar{C}'_i$. Subsequently, CS sends $\tilde{C}$ to AC. Because clusters are essential building blocks of our system model, the relationship between them is equivalent. Furthermore, one CS is in charge of one cluster. Therefore, some cloud servers failure does not influence other cloud servers, and the whole system still works fine. Therefore, PPMDP can support fault tolerance of user failures, fog device failures, and cloud server failures.

B. SYSTEM SCALABILITY

According to the above discussion, PPMDP can still work well in the case of some users leaving the system. Therefore, in this subsection, we focus on the case of user addition. When some new users add to the scope of the FD_i, the n_i will increase. If n_i is still less than or equal to n, FD_i can work normally. If $n_i > n$ and the number of FD of the corresponding cluster is less than m, let the number of users belong FD_i be n, and add a new FD into the cluster. Otherwise, add a new CS (that is, a new cluster) into the system, and add a new FD into the new cluster. The number of users of the new FD is $n_i - n$. For AC, there is not an upper bound on the number of CS. Therefore, our protocol PPMDP has pretty good system scalability.

IX. PERFORMANCE ANALYSIS

Our protocol PPMDP and the protocol presented by [31] both can achieve complex functions' privacy-preserving computation by breaking the link between data and its source. Nevertheless, the specific implementation methods are different. Suppose there are n users, and the data range is $[0, 2^l]$. In [31], each user needs to encrypt an ln-bit string to a random ln-bit string by leveraging bitwise-XOR homomorphic encryption. In our protocol PPMDP, each user encrypts his data by utilizing the Paillier encryption to obtain a 2048-bit ciphertext (when the security parameter $\kappa = 512$). Because [31] uses efficient XOR operation to realize encryption and decryption, the protocol presented by [31] has better computation efficiency. However, it is not suitable for the application scenarios at scale. The total communication overheads of [31] and PPMDP are ln^2 bits and $2048n$ bits, respectively. Since the number of FDs is tiny relative to the number of users, we ignore the aggregation ciphertexts in PPMDP's communication overhead. As the number of users increases, the growth rate of PPMDP computational cost is linear order, but that of the protocol presented by [31] is square order. Therefore, PPMDP is more suitable for massive users' application scenarios.

There is another significant difference between the two protocols. That is the way to realize random permutations securely. In [31], there is a trusted authority that decides the random permutation and distributes it to users, which means each round of data submission requires the trusted authority's participation. Nevertheless, there is not a trusted authority in our protocol PPMDP, and honest-but-curious FDs and CS dynamically pick the random permutations. According to these random permutations, PPMDP utilizes the Paillier encryption and Horner's Rule to securely permute users' meter data.

Next, we discuss the performance of our protocol PPMDP under the system settings as follows: η clusters, m groups in each cluster, n users in each group. In other words, there is a total of ηmn users. We use a δ-digit ternary number to represent each user's meter data. Compared to the exponentiation operation, the computational cost of multiplication operation in $\mathbb{Z}_N^*$ is negligible [34]. Because the primary operations of Alg.1 are modular and division operation over integer fields, the computational cost of Alg.1 is also negligible. T_e and T_m denote the computational costs of a modular exponential operation in $\mathbb{Z}_{N^2}^*$ and a modular multiplication operation, respectively. Let the security parameter $\kappa = 512$. We adopt the experiment environment of the reference [18]. OpenSSL Library [35] is run on a 2.65-GHz processor 4GB-memory computing machine to estimate the computational costs of operations. According to [18], $T_e = 10.082$ms, $T_m = 0.016$ms. According to the Section VI, we set $n = 100$, $m = 5$. That is, the maximum number of users of one cluster is $mn = 500$. Note that if a system has $1,000,000$ smart meters, because the size of a cluster is 500 and one cloud server manages one cluster under the above system setting, there is a need for 2000 cloud servers and many more fog devices, somewhat unsuitable. However, in practical deployment, a cloud server can be in charge of several clusters according to the cloud server's ability, and analogously, a fog device can manage several groups. Therefore, in practical deployment, then PPMDP needs a limited number of cloud servers and fog devices. For example, for $1,000,000$ smart meters, if a cluster's size is $100,000$, one cloud server manages 5 clusters, PPMDP needs merely 2 cloud servers. Besides, if the size of a group is $10,000$, PPMDP needs merely 100 fog devices.

A. COMPUTATIONAL COSTS ANALYSIS

In PPMDP, each user performs 2δ times modular exponential operations and δ times modular multiplication operations; and each FD executes $\delta n+\delta$ times modular exponential operations and $2\delta(n-1)$ times modular multiplication operations; and each CS accomplishes δm times modular exponential operations and $2\delta(m-1)$ times modular multiplication operations. η clusters execute PPMDP in parallel. After one-round communication, AC obtains all data that had been perturbed. Because the data dealt with by AC is plaintexts, here we do not consider its computational cost. Therefore, the total computational cost of PPMDP is the computational cost in one-round communication, as shown in Table 2. It is $107\delta T_e + (105\delta - 1)T_m$ and is in proportion to δ. At the same time, δ determines the upper bound W of a meter data. Fig.3 illustrates the effect of δ on W and PPMDP's computational cost, which can help us choose the suitable parameter δ according to the concrete application scenarios. When $\delta = 3$, the value range of a meter data is $[0, W = 26]$, W is too small for practical application scenarios. When $\delta = 4$, the data range is $[0, W = 80]$, the total computational cost of PPMDP is 4.3s for 500η users; $\delta = 5$, the data range is $[0, W = 242]$, the total computational cost is 5.4s; $\delta = 6$, the data range is $[0, W = 728]$, the total computational cost is 6.5s; and $\delta = 7$, the data range is $[0, W = 2186]$, the total computational cost is 7.6s. Because a single user's meter data is not too large in each time slot, it is reasonable to assume under normal conditions, the individual's electricity consumption data in a time slot is usually less than 2186. Therefore, $\delta = 4$ or 5 or 6 or 7 can satisfy different practical application scenarios. When $\delta = 4$, the computational costs of User, FD, and CS are 0.081s, 4.086s, and 0.202s, respectively. When $\delta = 7$, the computational costs of User, FD, and CS are 0.14s, 7.068s, and 0.353s, respectively. The computational cost of FD is 93.5% of the total computational cost. Note that if the hardware performance of FD is better than that of our experimental environment, PPMDP's computational cost will be reduced significantly. Besides, we must note that PPMDP's computational cost is relative to a cluster's size and not to the number of clusters. Therefore, PPMDP is suitable for smart grids.

TABLE 2. Computational costs of our protocol PPMDP.

User	FD	CS
$2\delta T_e + \delta T_m$	$\delta n T_e + \delta(n-1)T_m$	$\delta m T_e + \delta(m-1)T_m$

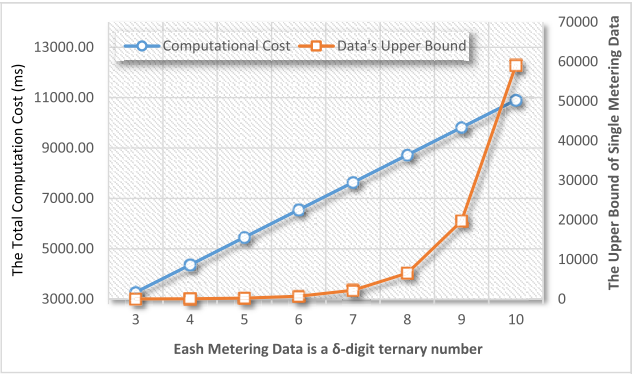

FIGURE 3. Effect of δ on meter data scope W and PPMDP's computational cost.

Following, we compare the computational costs of our protocol PPMDP with MuDA [16] and PPM-HDA [17]. Note that, for the sake of simple description, the number of users in the three protocols is denoted as *Num*. Besides, because differential privacy is beyond the scope of this paper, we ignore the performance overhead of differential privacy of MuDA and PPM-HDA. Also, need to pay attention to that PPM-HDA consists of two protocol, MHDA$^+$ (which realizes privacy-preserving additive aggregate functions) and MHDA$^{\oplus}$ (which achieves privacy-preserving non-additive aggregate functions). For convenience sake, regarding non-additive functions, here we only consider the performance costs of minimum function. T_b denotes the computational costs of a bilinear pairing operation, respectively. T_p refers to the computational costs of using Pollard's lambda method to compute the discrete logarithm. T_p is directly

proportional to the number of users and the bit length of a message. In the above-mentioned experiment environment, $T_b = 21.823ms$. Set the bit length of a message is 13 and the number of users is 10, 000, then $T_p = 42.875$ms. We set *Num* to be an integer multiple of 10, 000. For simplicity, T_p always takes the value 42.875ms. Because of the bit length of a message is 13, we need a 9-digit ternary number to express single meter data in our protocol PPMDP. Besides, in PPMDP, when $|N| = 1024$, we set $n = 100$ and $m = 5$. That is, the maximum number of users of one cluster is 500. Therefore, when $Num = 10,000$, we let $\eta = 20$; when $Num = 20,000$, we let $\eta = 40$, and so on.

The computational costs required for MuDA to achieve the security calculation of average and variance are shown in Table 3. The computational costs for PPM-HDA to perform the security computation of average and minimum are shown in Table 4. In Table 4, d is the degree of the secret polynomial function of PPM-HDA. Here we set d equal to 2. In PPM-HDA, each user first utilizes the prefix membership verification protocol [36], [37] to compute its prefix family, in which each prefix represents a range containing its data. Then the user transforms each prefix of its prefix family into a unique binary number by using the prefix numeralization protocol [38]. After that, the user encrypts these binary numbers and sends the corresponding ciphertexts to cloud servers. Here, for PPM-HDA, we only consider the computational cost of the user doing encryption operation but ignore the calculation overhead of the user doing other operations. In our protocol PPMDP, after one-round communication, AC obtains all data that had been perturbed. So the calculating cost for PPMDP to realize the security computation of average, variance, and the minimum is the computational cost in one-round communication. The computational cost is $107\delta T_e + (105\delta - 1)T_m$, here $\delta = 9$.

TABLE 3. Computational costs of MuDA [16].

	User	GW	CC
average	$2T_e + T_m$	$(Num-1)T_m$	$T_e + T_p$
variance	$2T_e + T_m$	$3(Num-1)T_m + Num \cdot T_b$	$2(T_e + T_p)$

TABLE 4. Computational costs of PPM-HDA [17].

	User	SP	CS
average (MHDA$^+$)	$2T_e + T_m$	$(Num-1)T_m$	$(d+1)T_e + d \cdot T_m + T_p$
minimum (MHDA$\oplus$)	$2T_e + T_m$	0	$(d+1)T_e + d \cdot T_m + Num \cdot T_p$

According to Tables 3 and 4, we can obtain the computational costs of security calculation of average function of three schemes, the calculation costs of security calculation of variance function of MuDA and PPMDP, and the computational costs of security calculation of minimum function of PPM-HDA and PPMDP, as shown in Figs.4, 5 and 6, respectively. Fig.4 shows that the computational costs of MuDA and PPM-HDA(MHDA$^+$) are very close and less than

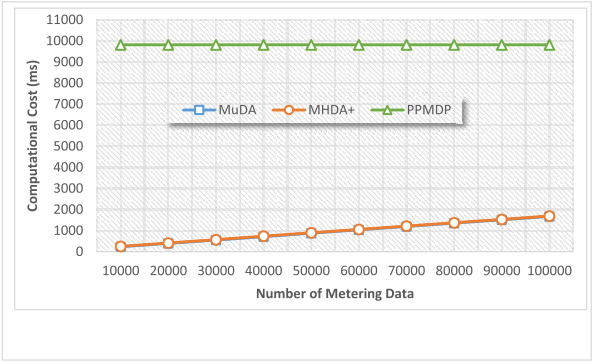

FIGURE 4. Comparison of computational costs of average function.

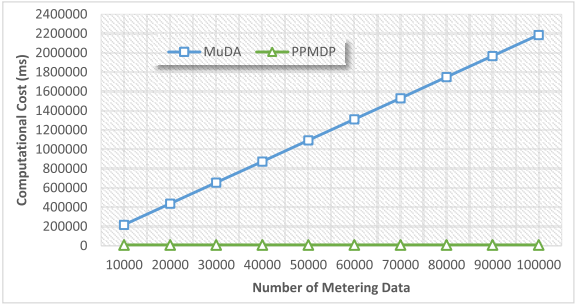

FIGURE 5. Comparison of computational costs of variance function.

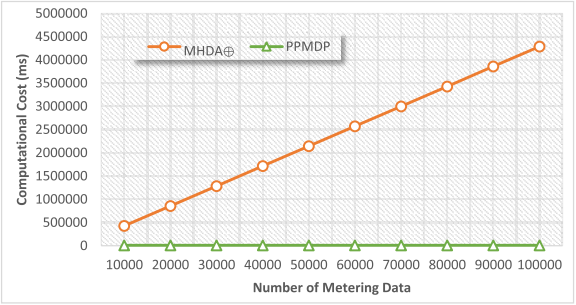

FIGURE 6. Comparison of computational costs of minimum function.

that of our protocol PPMDP. So PPMDP is not dominant when implementing security computation of simple average function. It is because, in MuDA or PPM-HDA(MHDA$^+$), GW or SP only need to perform $Num - 1$ times modular multiplication operations, but in our protocol PPMDP, FD and CS need to execute $n + m$ times modular exponential operations to achieve two-level data disturbance. However, when implementing security computation of complex variance function, the advantage of PPMDP is very obvious in relation to MuDA (as shown in Fig.5). MuDA needs to perform $3(Num - 1)$ times modular multiplication operations and Num times bilinear pairing operations to realize security calculation of variance, nevertheless PPMDP still only needs to perform $n + m$ times modular exponential operations. Furthermore, Fig.6 shows that when implementing

TABLE 5. Communication overhead of MuDA [16].

	User to GW	GW to CC
average	$Num \cdot L_{C_2}$	L_{C_2}
variance	$Num \cdot L_{C_2}$	$2L_{C_2}$

TABLE 6. Communication overhead of PPM-HDA [17].

	User to SP	SP to CS
average(MHDA^{+})	$Num \cdot L_{C_2}$	L_{C_2}
minimum(MHDA$^{\oplus}$)	$Num \cdot L_{C_2}$	$Num \cdot L_{C_2}$

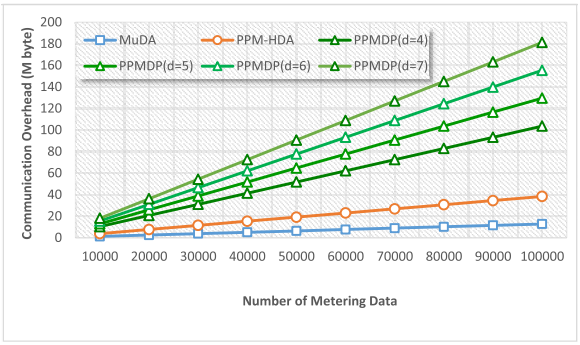

FIGURE 7. Comparison of communication overhead.

security calculation of minimum function PPMDP also has significant advantage in terms of computation performance relative to PPM-HDA(MHDA$^{\oplus}$). The reason is that in PPM-HDA(MHDA$^{\oplus}$), SP does not perform the aggregation operation on *Num* ciphertexts, which causes CS needs to call *Num* times Pollard's lambda method, which takes computational costs of $Num \cdot T_p$. However, PPMDP still only needs to perform $n + m$ times modular exponential operations to disturb data. Therefore, based on the above comparison analysis, it is clear that our protocol PPMDP performs better in terms of computational performance of realizing privacy-preserving calculation of complex functions and non-additive functions. The parallel execution between clusters brings its high-efficient computational performance.

B. COMMUNICATION OVERHEAD ANALYSIS

If the security parameter $\kappa = 512$, the length of each ciphertext of PPMDP is $L_{C_1} = 2048$ bits, and that of MuDA and PPM-HDA are $L_{C_2} = 1024$ bits. The communication overheads required for MuDA to achieve the security calculation of average and variance are shown in Table 5. The communication overhead for PPM-HDA to perform the security computation of average and minimum are shown in Table 6. According to Tables 5 and 6, the total communication overhead of MuDA to achieve security calculation of two functions is $(Num+3) \cdot L_{C_2}$, and that of PPM-HDA to realize security computation of two functions is $(3Num+1) \cdot L_{C_2}$. The total communication overhead of PPMDP to achieve security computation of multiple functions is $\eta\delta(mn+m+1) \cdot L_{C_1} = 2\eta\delta(mn+m+1) \cdot L_{C_2}$. Note, in PPMDP, when $|N| = 1024$, we set $n = 100$ and $m = 5$. Therefore, when $Num = 10,000$, we let $\eta = 20$; when $Num = 20,000$, we let $\eta = 40$, and so on. From Fig.7, we can observe when the number of data is $100,000$, the communication overhead of MuDA is about $12.8M$ bytes, that of PPM-HDA is about $38.4M$ bytes, that of PPMDA is about $103.6M$ bytes when $\delta = 4$ and is about $181.3M$ bytes when $\delta = 7$. The communication overhead of PPMDA is higher than that of PPM-HDA and that of MuDA. However, we should note that the communication overheads of PPM-HDA and MuDA are proportional to the number of functions. In contrast, the number of functions does not influence PPMDP's communication overhead. Therefore, in terms of communication overhead, when processing lots of functions simultaneously, our protocol PPMDP has certain advantages.

X. CONCLUSION

This paper proposed a privacy-preserving multi-functional data processing protocol (named PPMDP) for smart grids. It can achieve a random permutation of massive meter data in a privacy-preserving way, which thanks to properties of the Paillier homomorphic encryption and Horner's Rule and a two-level permutation mechanism. Because AC could obtain the raw meter data (but not know the source of each data), it could realize efficient and secure calculation of plenty of functions (such as variance, comparing, linear regression analysis) of extensive meter data simultaneously. The security analysis showed that PPMDP ensures data confidentiality and data source anonymity; the feature analysis revealed that PPMDP is pretty good scalable. Performance analysis illustrated that PPMDP is highly efficient in terms of computational costs and communication overhead. The current PPMDP is not suitable for positioning applications, such as locating the users who consume tremendous electricity. Hence, for future works, we will improve the current work to overcome the above application limitation. Meanwhile, we will extend the current work to multi-dimensional data and study other security issues (e.g., internal adversary attack, differential attacks). We try to develop effective protocols to realize secure multi-functional multi-dimensional data processing and resist more attacks. Furthermore, the current work only considers the privacy-preserving processing of users' electricity consumption data. However, in smart grids, households equipped with mini electricity generators (e.g., solar panels) can inject electricity into the electrical grid [1]. Therefore, there is another direction for our future work. We will study and design privacy-preserving meter data processing protocols or schemes for electricity consumption data and electricity injected into the electrical grid simultaneously.

REFERENCES

[1] M. A. Mustafa, S. Cleemput, A. Aly, and A. Abidin, "A secure and privacy-preserving protocol for smart metering operational data collection," *IEEE Trans. Smart Grid*, vol. 10, no. 6, pp. 6481–6490, Nov. 2019.

[2] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, "PPFA: Privacy preserving fog-enabled aggregation in smart grid," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3733–3744, Aug. 2018.
[3] H. Shen, Y. Liu, Z. Xia, and M. Zhang, "An efficient aggregation scheme resisting on malicious data mining attacks for smart grid," *Inf. Sci.*, vol. 526, pp. 289–300, Jul. 2020.
[4] J. Wang, L. Wu, K.-K.-R. Choo, and D. He, "Blockchain-based anonymous authentication with key management for smart grid edge computing infrastructure," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1984–1992, Mar. 2020.
[5] L. Zhu, M. Li, Z. Zhang, C. Xu, R. Zhang, X. Du, and N. Guizani, "Privacy-preserving authentication and data aggregation for fog-based smart grid," *IEEE Commun. Mag.*, vol. 57, no. 6, pp. 80–85, Jun. 2019.
[6] D. Abbasinezhad-Mood and M. Nikooghadam, "Efficient design and extensive hardware evaluation of an anonymous data aggregation scheme for smart grid," *Secur. Privacy*, vol. 1, no. 2, p. e24, Mar. 2018.
[7] C. Ge, C. Yin, Z. Liu, L. Fang, J. Zhu, and H. Ling, "A privacy preserve big data analysis system for wearable wireless sensor network," *Comput. Secur.*, vol. 96, Sep. 2020, Art. no. 101887.
[8] M. Zhang, Y. Chen, and W. Susilo, "PPO-CPQ: A privacy-preserving optimization of clinical pathway query for E-healthcare systems," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10660–10672, Oct. 2020.
[9] Q. Jiang, N. Zhang, J. Ni, J. Ma, X. Ma, and K.-K.-R. Choo, "Unified biometric privacy preserving three-factor authentication and key agreement for cloud-assisted autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9390–9401, Sep. 2020.
[10] Q. Jiang, Z. Chen, J. Ma, X. Ma, J. Shen, and D. Wu, "Optimized fuzzy commitment based key agreement protocol for wireless body area network," *IEEE Trans. Emerg. Topics Comput.*, early access, Oct. 23, 2019, doi: 10.1109/TETC.2019.2949137.
[11] H. Shen, M. Zhang, and J. Shen, "Efficient privacy-preserving cube-data aggregation scheme for smart grids," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 6, pp. 1369–1381, Jun. 2017.
[12] H. Shen, M. Zhang, H. Wang, F. Guo, and W. Susilo, "A lightweight privacy-preserving fair meeting location determination scheme," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3083–3093, Apr. 2020.
[13] Y. Liu, W. Guo, C.-I. Fan, L. Chang, and C. Cheng, "A practical privacy-preserving data aggregation (3PDA) scheme for smart grid," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1767–1774, Mar. 2019.
[14] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, "A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT," *IEEE Access*, vol. 5, pp. 3302–3312, 2017.
[15] J. Song, Y. Liu, J. Shao, and C. Tang, "A dynamic membership data aggregation (DMDA) protocol for smart grid," *IEEE Syst. J.*, vol. 14, no. 1, pp. 900–908, Mar. 2020.
[16] L. Chen, R. Lu, Z. Cao, K. AlHarbi, and X. Lin, "MuDA: Multifunctional data aggregation in privacy-preserving smart grid communications," *Peer-Peer Netw. Appl.*, vol. 8, no. 5, pp. 777–792, Sep. 2015.
[17] S. Han, S. Zhao, Q. Li, C.-H. Ju, and W. Zhou, "PPM-HDA: Privacy-preserving and multifunctional health data aggregation with fault tolerance," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 9, pp. 1940–1955, Sep. 2016.
[18] H. Ren, H. Li, X. Liang, S. He, Y. Dai, and L. Zhao, "Privacy-enhanced and multifunctional health data aggregation under differential privacy guarantees," *Sensors*, vol. 16, no. 9, p. 1463, Sep. 2016.
[19] M. Zhang, Y. Chen, Z. Xia, J. Du, and W. Susilo, "PPO-DFK: A privacy-preserving optimization of distributed fractional knapsack with application in secure footballer configurations," *IEEE Syst. J.*, vol. 15, no. 1, pp. 759–770, Mar. 2021, doi: 10.1109/JSYST.2020.2991928.
[20] H. Shen, M. Zhang, H. Wang, F. Guo, and W. Susilo, "A cloud-aided privacy-preserving multi-dimensional data comparison protocol," *Inf. Sci.*, vol. 545, pp. 739–752, Feb. 2021.
[21] Z. Guan, Y. Zhang, L. Wu, J. Wu, J. Li, Y. Ma, and J. Hu, "APPA: An anonymous and privacy preserving data aggregation scheme for fog-enhanced IoT," *J. Netw. Comput. Appl.*, vol. 125, pp. 82–92, Jan. 2019.
[22] S. M. A. Oteafy and H. S. Hassanein, "IoT in the fog: A roadmap for data-centric IoT development," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 157–163, Mar. 2018.
[23] M. Zhang, Y. Chen, and J. Huang, "SE-PPFM: A searchable encryption scheme supporting privacy-preserving fuzzy multikeyword in cloud systems," *IEEE Syst. J.*, early access, Jun. 11, 2020, doi: 10.1109/JSYST.2020.2997932.
[24] C. Ge, W. Susilo, Z. Liu, J. Xia, P. Szalachowski, and F. Liming, "Secure keyword search and data sharing mechanism for cloud computing," *IEEE Trans. Dependable Secure Comput.*, early access, Jan. 3, 2020, doi: 10.1109/TDSC.2020.2963978.
[25] Z. Shi, R. Sun, R. Lu, L. Chen, J. Chen, and X. Sherman Shen, "Diverse grouping-based aggregation protocol with error detection for smart grid communications," *IEEE Trans. Smart Grid*, vol. 6, no. 6, pp. 2856–2868, Nov. 2015.
[26] K. Kenthapadi, I. Mironov, and A. G. Thakurta, "Privacy-preserving data mining in industry," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, Melbourne, VIC, Australia, Jan. 2019, pp. 840–841, doi: 10.1145/3289600.3291384.
[27] G. Couteau, "New protocols for secure equality test and comparison," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Secur.* Cham, Switzerland: Springer, 2018, pp. 303–320.
[28] H. Zhu, F. Wang, R. Lu, F. Liu, G. Fu, and H. Li, "Efficient and privacy-preserving proximity detection schemes for social applications," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2947–2957, Aug. 2018.
[29] M. A. Mustafa, N. Zhang, G. Kalogridis, and Z. Fan, "DEP2SA: A decentralized efficient privacy-preserving and selective aggregation scheme in advanced metering infrastructure," *IEEE Access*, vol. 3, pp. 2828–2846, 2015.
[30] H. Mahdikhani, S. Mahdavifar, R. Lu, H. Zhu, and A. A. Ghorbani, "Achieving privacy-preserving subset aggregation in fog-enhanced IoT," *IEEE Access*, vol. 7, pp. 184438–184447, 2019.
[31] Y. Zhang, Q. Chen, and S. Zhong, "Privacy-preserving data aggregation in mobile phone sensing," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 5, pp. 980–992, May 2016.
[32] Y. Liu and Q. Zhao, "E-voting scheme using secret sharing and K-anonymity," *World Wide Web*, vol. 22, no. 4, pp. 1657–1667, Jul. 2019.
[33] M. Zhang, W. Song, and J. Zhang, "A secure clinical diagnosis with privacy-preserving multiclass support vector machine in clouds," *IEEE Syst. J.*, early access, Oct. 14, 2020, doi: 10.1109/JSYST.2020.3027758.
[34] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "EPPA: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 9, pp. 1621–1631, Sep. 2012.
[35] *OpenSSL 1.0.2. OpenSSL Software Foundation*. Accessed: Jul. 3, 2020. [Online]. Available: http://www.openssl.org/source/
[36] J. Cheng, H. Yang, S. H. Y. Wong, P. Zerfos, and S. Lu, "Design and implementation of cross-domain cooperative firewall," in *Proc. IEEE Int. Conf. Netw. Protocols*, Beijing, China, Oct. 2007, pp. 284–293, doi: 10.1109/ICNP.2007.4375859.
[37] F. Chen and A. X. Liu, "SafeQ: Secure and efficient query processing in sensor networks," in *Proc. IEEE INFOCOM*, San Diego, CA, USA, Mar. 2010, pp. 1–9, doi: 10.1109/INFCOM.2010.5462094.
[38] Y.-K. Chang, "Fast binary and multiway prefix searches for packet forwarding," *Comput. Netw.*, vol. 51, no. 3, pp. 588–605, Feb. 2007.

HUA SHEN received the Ph.D. degree in computer science from the School of Computer Science, Wuhan University, Wuhan, China, in 2014. She was a Visiting Fellow with the School of Computer and Information Technology, Institute of Cybersecurity and Cryptology, University of Wollongong, Wollongong, NSW, Australia, from January 2019 to January 2020. She is currently an Associate Professor with the School of Computers, Hubei University of Technology, Wuhan. Her research interests include technology of privacy preserving, information security, and secure cloud computing.

MINGWU ZHANG received the Ph.D. degree in computer science from the College of Mathematics and Informatics, South China Agricultural University, Guangzhou, China, in 2009. He was a Japan Society for the Promotion of Science Fellow with the Institute of Mathematics for Industry, Kyushu University, Fukuoka, Japan, from 2010 to 2012. From 2015 to 2016, he was a Senior Research Fellow with the Centre for Computer and Information Security, University of Wollongong, Wollongong, NSW, Australia. He is currently a Professor with the Hubei University of Technology, Wuhan, China. He is also a Researcher with the State Key Laboratory of Cryptology and a Professor with the Guilin University of Electronic Technology. His current research interests include cryptography technology for clouds and big data, and privacy preservation.

HAO WANG received the Ph.D. degree in computer science from Shandong University, China, in 2012. He is currently working as an Associate Professor with Shandong Normal University. He is also focusing on attribute-based cryptography, secure multi-party computation, and block-chain. His research interest includes public key cryptography, in particular, designing cryptographic primitives and provable security.

FUCHUN GUO received the B.S. and M.S. degrees from Fujian Normal University, China, in 2005 and 2008, respectively, and the Ph.D. degree from the University of Wollongong, Australia, in 2013. He is currently an Associate Research Fellow with the School of Computing and Information Technology, University of Wollongong. His research interests include public-key cryptography; in particular, protocols, encryption and signature schemes, and security proof.

WILLY SUSILO (Fellow, IEEE) received a Ph.D. degree in computer science from the University of Wollongong, Australia. He is currently a Professor and the Head of the School of Computing and Information Technology and the Director of the Institute of Cybersecurity and Cryptology (iC^2), University of Wollongong. His main research interest includes applied cryptography. He was previously awarded the prestigious ARC Future Fellow from the Australian Research Council (ARC).