An Efficient and Secure Multidimensional Data Aggregation for Fog-Computing-Based Smart Grid

Omar Rafik Merad-Boudia, Sidi Mohammed Senouci

▶ **To cite this version:**

Omar Rafik Merad-Boudia, Sidi Mohammed Senouci. An Efficient and Secure Multidimensional Data Aggregation for Fog-Computing-Based Smart Grid. IEEE Internet of Things Journal, 2021, 8 (8), pp.6143-6153. 10.1109/JIOT.2020.3040982 . hal-03323212

HAL Id: hal-03323212
https://hal.science/hal-03323212

Submitted on 11 Feb 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

An Efficient and Secure Multidimensional Data Aggregation for Fog-Computing-Based Smart Grid

Omar Rafik Merad-Boudia and Sidi Mohammed Senouci, *Member, IEEE*

***Abstract*—The secure multidimensional data aggregation (MDA) has been widely investigated in smart grid for smart cities. However, previous proposals use heavy computation operations either to encrypt or to decrypt the multidimensional data. Moreover, previous fault-tolerant mechanisms lead to an important computation cost, and also a high communication cost when considering a separate identification phase. In this article, we propose an efficient and secure MDA scheme, named ESMA. Unlike existing schemes, the multidimensional data in ESMA are structured and encrypted into a single Paillier ciphertext and thereafter, the data are efficiently decrypted. For privacy preserving, the Paillier cryptosystem is adopted in a fog computing-based architecture, and to achieve efficient authentication, the batch verification technique is applied. Besides, ESMA is fault tolerant, i.e., even if some of the smart meters fail to send their data, the final aggregation result will not be affected. Furthermore, ESMA can be adapted to respond to other queries than the summation of data. The performance analysis demonstrates the cost efficiency of ESMA both in computation and communication and the scalability as well. For instance, with a 16-bits size for each data type and 500 reporting smart meters, 40 data types can be supported in a single Paillier ciphertext. ESMA also resists various security attacks and preserves the user's privacy.**

***Index Terms*—Data aggregation, fault tolerance (FT), fog computing, security and privacy, smart cities, smart grid (SG).**

I. Introduction

THE SMART grid (SG) network is the next-generation power grid network, which utilizes modern information and communication technologies and enables two-way communication between customers and service providers. Many electric utility companies have already substituted or introduced SG alongside the traditional power grid [1]. This gives electric utility companies the capability to perform fault diagnosis and to considerably enhance their power generation, distribution, transmission, and control. This can also avoid power blackouts and give the ability to anticipate power demands and facilitate the integration of renewable energy technologies in the grid [2], [3]. In SG, utility companies use smart meters (SMs) to gather real-time consumption data and other information. The SM is considered as the most critical element placed at the user side. It enables customers to report their real-time consumption data (e.g., every quarter hour) to the control center (CC), and based on these data, the CC can anticipate power demands and adjust power generation, dynamic pricing, and so on. For instance, Kumari *et al.* [4] proposed a solution that accurately predicts future load consumption based on historical data using deep learning to save energy at the demand side and reduce energy production at the supply side of the system. However, SG is subject to several threats against security and privacy, e.g., by analyzing the real-time electricity usage data, an attacker would be able to threaten the user's privacy, and thus, inferring the user's life habits such as when a user exits the house. Consequently, some cryptographic techniques should be used to preserve the user's privacy [5].

The concept of data aggregation in the context of SG is introduced in [6]. A traditional data aggregation scheme considers a gateway node that aggregates the SMs reports and sends the result to the CC, enabling the utility company to get the total consumption data. This gives the company the capability to dynamically adjust power distribution according to these data and to protect users' privacy as well. Recently, a fog computing architecture has been considered in SG for efficient aggregation [7]. With the help of such architecture, Nazmudeen *et al.* [8] proposed a framework for a distributed data aggregation. As stated in [9], traditional in-network aggregation' architectures face resource and scalability problems. They employ less powerful intermediate nodes and a large number of end nodes are usually connected with one intermediate node. By considering fog nodes (FNs), we can exploit FN's intrinsic abilities in terms of communication, computation, and storage, leading, therefore, to the optimal use of these nodes for aggregation purposes. This can also resolve the bandwidth and latency issues that arise in a cloud computing architecture [7].

Although previous solutions [6], [9]–[19] can prevent the disclosure of sensitive information to the adversary or the utility company, there are, however, some unresolved issues. First, the data aggregated are likely to be multidimensional (electricity consumption record, time, consumption purpose, etc.). Hence, more data provided more efficient is the analysis. Some works only enable one-dimensional aggregation (ODA), i.e., every dimension is processed separately. Moreover, existing multidimensional aggregation (MDA) schemes use

Omar Rafik Merad-Boudia is with the Computer Science Department, University of Oran 1 Ahmed Ben Bella, Oran 31000, Algeria, and also with the STIC Laboratory, University of Tlemcen, Tlemcen 13000, Algeria (e-mail: rafik.merad@univ-oran1.dz).

Sidi Mohammed Senouci is with the DRIVE Laboratory, University of Burgundy, 58000 Nevers, France (e-mail: sidi-mohammed.senouci@u-bourgogne.fr).

complex cryptographic operations to encrypt or to decrypt the multidimensional data. Second, in some schemes, there is no mechanism for FT, in fact, in SG, SMs could malfunction for a certain amount of time and thus could not send their reports to the FN. In previous schemes, the CC has to ask FNs for the nonresponding SMs that leads to an important delay, and the aggregation process is then halted. Finally, previous schemes are only conceived to respond to the sum query. However, the CC may need other statistical functions than summation such as variance to further analyze power consumption. This could allow the detection of abnormal situations and load imbalance. Moreover, the CC may need to learn about the number of customers whose SM's reading is higher than a threshold, and also the total usage of these customers for predicting better power generation.

For these reasons, we propose ESMA, an Efficient and Secure Multidimensional data Aggregation for fog computing-based smart grid (FCSG). The main contributions of this article can be summarized as follows.

1) We propose an efficient MDA for FCSG. In ESMA, even if the data of each SM are multidimensional, the data reported to FN would still be a single Paillier ciphertext. Besides, we employ an encoding function that gives the CC the capability to efficiently recover the aggregated data from the ciphertext. The computation and communication resources are then conserved in comparison to previous schemes.
2) We employ the Paillier Cryptosystem and its homomorphic property to preserve user's privacy. So, the FN performs aggregation on ciphertexts, which prevents the disclosure of sensitive data. Also, we propose a fault-tolerant mechanism to ensure that the final aggregated data recovered at CC are correct even if some nodes are malfunctioning.
3) We illustrate how ESMA can be adapted to answer other queries than summation. We also provide security analysis and performance evaluation to show the security and the efficiency of ESMA in comparison with the previous work.

The remainder of this article is organized as follows. Section II introduces the related work. Section III presents the system models and design goals. Section IV describes the preliminaries of ESMA before presenting its technical details in Section V. Section VI presents how ESMA can respond to other queries and Section VII discusses security analysis. Finally, Sections VIII and IX give the performance evaluation and conclusion, respectively.

II. Related Work

In this section, we present an overview of traditional solutions to secure data aggregation in SG [10]–[14], and also the recently proposed ones in the context of FCSG [9], [16]–[19]. Introduced in [6], protecting the user's privacy via data aggregation has recently become one of the most attractive research topics. Li *et al.* [6] introduced secure in-network aggregation in the context of SG; they employ homomorphic encryption (HE) to avoid the disclosure of user's private data to intermediate aggregator nodes. Lu *et al.* [10] used Paillier HE and employed a super-increasing sequence to secure MDA, an additional exponentiation operation is computed for every dimension by SM. Boudia *et al.* [11] proposed an elliptic curve-based secure MDA using El Gamal HE with multiple public keys. Their scheme does not require complex cryptographic operations en route and leads to lower overhead. However, an additional elliptic curve scalar multiplication is computed for every dimension by SM. Ming *et al.* [12] and Zuo *et al.* [13] employed a super-increasing sequence with El Gamal HE to achieve MDA. In these two schemes, the CC has to solve the discrete logarithm problem (DLP) to retrieve the aggregated data. Badra and Zeadally [14] proposed a lightweight secure ODA using symmetric HE and Diffie–Hellman methods. Shen *et al.* [15] introduced the malicious data mining attack and proposed a secure ODA using Paillier HE and bilinear maps.

Recently, many solutions have been proposed in the context of FCSG. Lyu *et al.* [9] proposed a privacy-preserving fog-enabled aggregation scheme using one-time pad HE. The authors consider ODA and an FN that aggregates the collected data, decrypts, and then encrypts noisy aggregation. They propose a fault-tolerant mechanism to achieve resilience to node failures. However, an additional phase is required for this purpose, which leads to additional communication cost. Moreover, data integrity is not ensured in their scheme. Liu *et al.* [16] presented a fog-enabled aggregation scheme that employs a double trapdoor HE to encrypt one-dimensional data. Note that their scheme can achieve MDA by using a super-increasing sequence. However, similarly to the work in [10], an additional heavy computation cost will be added for every dimension. Okay *et al.* [17] proposed two efficient and lightweight secure aggregation protocols for FCSG. The authors employ Domingo-Ferrer HE and Paillier HE to encrypt ODA. They present mathematical models for these protocols and give extensive simulations and comparisons. Saleem *et al.* [18] proposed an efficient and privacy-preserving ODA for FCSG. The authors exploit a modified version of Paillier HE in order to encrypt the metering data and use MAC for DI. They propose a fault-tolerant mechanism to achieve resilience to node failures. However, the CC has to compute the discrete logarithm to retrieve the aggregated data. Furthermore, the fault-tolerant decryption cost increases with the increasing number of faulty SMs. Zhao *et al.* [19] proposed an ODA scheme for FCSG that achieves multifunctional statistics; the authors employ somewhat homomorphic encryption (SHE) to preserve the user's privacy. However, their proposal incurs high overheads in terms of communication and computation.

We compare in Table I the aforementioned works in terms of the data type (DT), the considered architecture (CA), the used technique (UT), fault tolerance (FT), source authentication (SA), and data integrity (DI). The authors of these works either consider data aggregation with one dimension [6], [9], [16]–[19] or use heavy computation operations to encrypt or to decrypt the multidimensional data [10]–[13]. Note that these schemes are only conceived to respond to the sum query. Also, the FT mechanisms proposed in some schemes, e.g., the scheme in [9], requires a separate

TABLE I
COMPARATIVE ANALYSIS OF PREVIOUS SCHEMES

Scheme	DT	CA	UT	FT	PP	SA	DI
Li et al. [6]	ODA	Cloud	Paillier	No	Yes	No	No
Lu et al. [10]	MDA	Cloud	Paillier	No	Yes	Yes	Yes
Merad et al. [11]	MDA	Cloud	ECEG	No	Yes	Yes	Yes
Ming et al. [12]	MDA	Cloud	ECEG	No	Yes	Yes	Yes
Zuo et al. [13]	MDA	Cloud	ElGamal	No	Yes	Yes	Yes
Badra et al. [14]	ODA	Cloud	Sym. HE	No	Yes	Yes	Yes
Sheng et al. [15]	ODA	Cloud	Paillier	No	Yes	Yes	Yes
Lyu et al. [9]	ODA	Fog	OTP	Yes	Yes	No	No
Liu et al. [16]	ODA	Fog	Trap. HE	No	Yes	No	No
Saleem et al. [18]	ODA	Fog	Paillier	Yes	Yes	Yes	Yes
Zhao et al. [19]	ODA	Fog	SHE	No	Yes	No	No

phase to identify the faulty SMs and thus incurs a high communication cost. To overcome these drawbacks, we introduce in ESMA an encoding function to efficiently structure and secure the multidimensional data. ESMA can be adapted to respond to other queries than the summation of data without additional overhead. Moreover, the proposed FT mechanism preserves network resources.

III. MODELS AND DESIGN GOAL

In this section, we define our network model, attacker model, and identify the design goal of ESMA.

A. Network Model

Similar to the previous work based on a fog computing architecture proposed for SG [7]–[9], [16]–[19], we consider in ESMA a three-level system. Fig. 1 shows a large number of SMs in the first level, FNs in the second level, and the CC in the third level. We consider that the CC covers k_2 FNs and each FN in turn covers k_1 SMs. The CC is responsible for collecting the users' data to anticipate power demands and to adjust power generation, dynamic pricing, and so on. Each $\mathrm{SM}_{ij}(i = 1, \ldots, k_1, j = 1, \ldots, k_2)$ collects the multidimensional data, performs cryptographic operations, and sends a report to the corresponding FN_j. Once received, the reports are homomorphically aggregated by FN_j. The result is then forwarded to the CC. After the reception of the FN_j's report, the CC performs decryption and gets the aggregation result. This can help the CC to make appropriate decisions. For generating the public/private keys used in the Paillier cryptosystem and other secret parameters, we consider a trust authority (TA), which is a fully reliable entity. After providing the corresponding secret parameters to all of the entities in the system, TA will no longer be required in the MDA process.

B. Attacker Model

In our attacker model, we consider that FNs and CC are honest-and-curious, i.e., they follow the protocol faithfully but are curious about the multidimensional data contained in the reports, e.g., by keeping all of the reported data to optimize the chance to retrieve user's private data. Besides, we consider that the users are honest and will not report false consumption data. However, there exists an external adversary $\mathcal{A}$ that can eavesdrop the communication flows and try to identify the report's content. $\mathcal{A}$ can also launch active attacks, such as data modification, false data injection, replay, or stealing the user's private data by intruding the database of CC and FN.

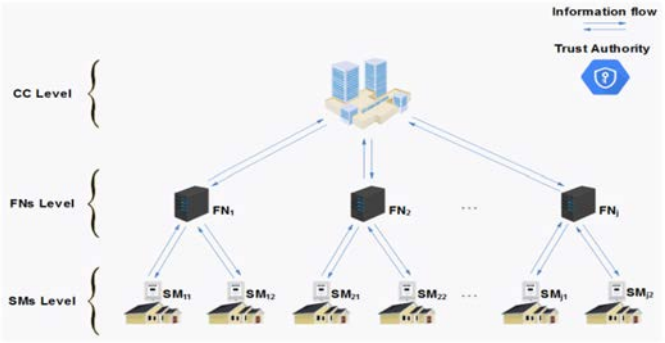

Fig. 1. System model.

In addition to the aforementioned attacks and due to the SM wastage, we also assume that certain SMs could be in failure status and thereby will fail to send their data to FN.

C. Design Goal

Our design goal is to propose a fault-tolerant, efficient, and privacy-preserving MDA scheme for FCSG such that the CC can obtain multiple real-time data from one single aggregated data. Specifically, the following goals should be satisfied.

1) The user's private data cannot be compromised by $\mathcal{A}$. The CC is the only entity that can read the aggregated multidimensional data in ESMA, and no one can have access to the sensitive individual data, including the CC.
2) FNs and CC should be able to verify the authenticity of the received reports. They should also detect if a report has been modified during the transmission and comes really from the legal entity.
3) The proposed scheme should be fault tolerant. When certain SMs fail to communicate their data, they should be detected by the corresponding FN and reported to the CC.
4) The computation cost of cryptographic operations performed at SM, FN, and CC should be efficient. Moreover, the multidimensional data should be encrypted into one single ciphertext to reduce the cost of communication.

IV. PRELIMINARIES

In this section, we briefly introduce the security techniques used in ESMA, such as the Paillier homomorphic cryptosystem and bilinear maps.

A. Paillier Homomorphic Cryptosystem

The Paillier homomorphic cryptosystem is a classic encryption [20] that can achieve additive homomorphism property. The encryption is proved to be semantically secure against chosen-plaintext attacks. Thus, it has been widely exploited in secure data aggregation protocols (see Table I). The key generation, encryption, and decryption processes operate as follows.

1) *Key Generation:* Given a security parameter κ, two κ bit prime numbers p_0 and q_0 are randomly selected. Let the

TABLE II
NOTATIONS

Notation	Definition
ODA	One-dimensional aggregation
MDA	Multidimensional aggregation
CC	The control center
FN	The fog node
SM	The smart meter
DLP	Discrete Logarithm Problem
ECDLP	Elliptic Curve Discrete Logarithm Problem
n	The modulus n = p_0q_0
g	The generator of $\mathbb{Z}_{n^2}$
$\mathbb{G}_1$	An additive group
$\mathbb{G}_2$	A multiplicative group
(n, g)	The public key pair
(λ, μ)	The private key pair
e	A bilinear pairing
H	A secure hash function, H: $\{0,1\}^* \rightarrow \mathbb{G}_1$
k_1	The number of SM covered by one FN
k_2	The number of FN covered by CC
x_{ij}	The secret share of SM_{ij}
sk_{ij}	The secret key of SM_{ij}
pk_{ij}	The public key of SM_{ij}
m_{il}	The data type l of SM_{ij}
d_{il}	The encoded form of m_{il}
l	The number of data types
2^z	The maximum value of a data type

RSA modulus $n = p_0q_0$ and $\lambda = lcm(p_0 - 1, q_0 - 1)$.We define a function $L(u) = u - 1/n$ and calculate $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$, where g is the generator of $\mathbb{Z}^*_{n^2}$ such that $gcd(L(g^\lambda \bmod n^2), n) = 1$. The public key is the tuple (n, g) and the corresponding private key is the tuple (λ, μ).

2) *Encryption:* Choose a random number $r \in \mathbb{Z}^*_n$ to encrypt a given message $m \in \mathbb{Z}_n : c = E(m) = g^m \cdot r^n \bmod n^2$.
3) *Decryption:* For the ciphertext c, the corresponding plaintext is calculated by $m = D(c) = L(c^\lambda \bmod n^2)\ \mu \bmod n$.

B. Bilinear Maps

Let $\mathbb{G}_1$ be an additive group and $\mathbb{G}_2$ be a multiplicative group, both of prime order q, i.e., $|\mathbb{G}_1| = |\mathbb{G}_2| = q$. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 = \mathbb{G}_2$, which has the following properties.

1) *Bilinearity:* $\forall P, Q \in \mathbb{G}_1$ and $\forall a, b \in \mathbb{Z}^*_q$, $e(aP, bQ) = e(P, Q)^{ab}$
2) *Non Degeneracy:* $\exists P \in \mathbb{G}_1$ where $e(P, Q) \neq 1_{\mathbb{G}2}$
3) *Computability:* $\forall P, Q \in \mathbb{G}_1$, there exists an efficient algorithm to compute e (P, Q) in polynomial time.

We use the notation Gen to denote the algorithm, taken as input a security parameter κ_1, which outputs a 5-tuple $(q, P, \mathbb{G}_1, \mathbb{G}_2, e)$, where q is κ_1-bit prime, $\mathbb{G}_1$ and $\mathbb{G}_2$ are two cyclic groups with order q, $P \in \mathbb{G}_1$ is a generator, and e is a nondegenerated and computable bilinear map.

V. Our Proposed ESMA Scheme

In this section, we present ESMA for FCSG; it mainly consists of five main parts: 1) system initialization; 2) registration; 3) SM report generation; 4) data aggregation; and 5) data reading. A list of acronyms and symbols used in this article along with their meaning is shown in Table II.

A. System Initialization

For the fog-based SG system under consideration, we suppose that TA bootstraps the whole system and in charge of system parameters configuration. The parameter generation and the secret shares generation processes operate as follows.

1) *Parameter Generation:* Besides producing two security parameters (κ, κ_1), as mentioned in Section IV, the TA randomly selects two κ-bit large prime numbers p_0 and q_0 and computes $n = p_0q_0$ and $\lambda = lcm(p_0 - 1, q_0 - 1)$. Then, the TA constructs a function $L(u) = u - 1/n$ and calculates $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$, where g is the generator of $\mathbb{Z}^*_{n^2}$ such that $gcd(L(g^\lambda \bmod n^2), n) = 1$. The public key is the tuple (n, g) and the corresponding private key is the tuple (λ, μ). Furthermore, given κ_1, TA generates the bilinear parameters $(q, P, \mathbb{G}_1, \mathbb{G}_2, e)$ by running Gen(κ_1). Then, TA defines a secure cryptographic hash function $H : \{0, 1\}^* \rightarrow \mathbb{G}_1$.
2) *Secret Shares Generation:* Using a pseudorandom number generator function, TA generates $k_1 * k_2$ secret shares $x_{ij} \in \mathbb{Z}^*_n$, $(i = 1, \ldots, k_1, j = 1, \ldots, k_2)$ and computes x_{0j} such that

$$x_{0j} = \sum_{i=1}^{k_1} x_{ij} \bmod n, \text{ where } j = 1, \ldots, k_2. \quad (1)$$

Note that the adversary cannot calculate x_{0j} without knowing all secret shares x_{ij}.

B. Registration

All SM_{ij}, FN_j, and CC need to be registered in the TA, we illustrate this phase as follows.

1) *Registration of SM_{ij}:* The SM_{ij} (for all $i \in \{1, 2, \ldots, k_1\}$) first chooses an identity ID_{ij}. Then, TA randomly chooses a number $sk_{ij} \in \mathbb{Z}^*_q$ and computes $pk_{ij} = sk_{ij}P$, and securely delivers sk_{ij} to SM_{ij}. Moreover, TA transmits the secret x_{ij}.
2) *Registration of FN_j:* The FN_j (for all $j \in \{1, 2, \ldots, k_2\}$) first chooses an identity ID_j. Then, TA randomly chooses a number $sk_j \in \mathbb{Z}^*_q$ and computes $pk_j = sk_jP$. Finally, the TA securely delivers sk_j to FN_j.
3) *Registration of CC:* The CC first chooses an identity ID_{CC}. Then, TA transmits the tuple (λ, μ), and the secret shares x_{ij} along with x_{0j}.

At the end of this phase, TA publishes the system parameters as $\{e, q, n, g, P, pk_{ij}, pk_j, \mathbb{G}_1, \mathbb{G}_2, H\}$.

C. Smart Meter Report Generation

As the client's consumption information is periodically reported to FN, e.g., every quarter hour, to protect consumer privacy from exposure, an SM needs to encrypt such private information. Thus, each SM_{ij} measures and generates its l types of data $(m_{i1}, m_{i2}, \ldots, m_{il})$. The following specific steps are performed.

Step-1: Let z be the maximum number of bits that could represent a DT m, SM_{ij} encodes its l types of data $(m_{i1}, m_{i2}, \ldots, m_{il})$ into $(d_{i1}, d_{i2}, \ldots, d_{il})$ and constructs

$\overline{D_{ij}}$ as follows:

$$d_{ik} = (m_{ik})_2 \| 0^{\theta}, \ k = 1, \ldots, l \tag{2}$$
$$\text{where } \theta = \left(\lceil \log_2(k_1) \rceil + z\right) * (k-1)$$
$$D_{ij} = d_{i1} + d_{i2} + \cdots + d_{il}. \tag{3}$$

After encoding, SM_{ij} computes $\overline{D_{ij}}$

$$\overline{D_{ij}} = D_{ij} + x_{ij} \bmod n. \tag{4}$$

Step-2: SM_{ij} chooses a random number $r_{ij} \in \mathbb{Z}_n^*$, and computes the ciphertext as follows:

$$C_{ij} = g^{\overline{D_{ij}}} \cdot r_{ij}^n \bmod n^2. \tag{5}$$

Step-3: SM_{ij} uses its private key sk_{ij} to compute the signature as follows:

$$\sigma_{ij} = sk_{ij} H\left(C_{ij} \| \text{ID}_{ij} \| \text{TS}\right) \tag{6}$$

where TS is the current timestamp.

Step-4: SM_{ij} sends, to the corresponding FN_j, the data packet that contains $\{C_{ij}, \text{ID}_{ij}, \text{TS}, \sigma_{ij}\}$. It is necessary to stress that every SM_{ij} should produce and encrypt a message D_{ij} as above. An example with $l = 3$ is depicted in Fig. 2.

D. Data Aggregation

This part consists of two possible phases. In the case if all of the SM_{ij} send their reports, FN_j will perform an *all-inclusive data aggregation*; otherwise, it will perform a *fault-tolerant data aggregation*.

1) All-Inclusive Data Aggregation: If all of the SM_{ij} report their data to the corresponding FN_j, FN_j will first perform the batch verification to verify the received signatures, i.e., check whether

$$e\left(P, \sum_{i=1}^{k_1} \sigma_{ij}\right) = \prod_{i=1}^{k_1} e\left(pk_{ij}, H\left(C_{ij} \| \text{ID}_{ij} \| \text{TS}\right)\right). \tag{7}$$

If the equation does hold, it means the signatures are valid. The correctness is as follows:

$$\begin{aligned} e\left(P, \sum_{i=1}^{k_1} \sigma_{ij}\right) &= e\left(P, \sum_{i=1}^{k_1} sk_{ij} H\left(C_{ij} \| \text{ID}_{ij} \| \text{TS}\right)\right) \\ &= \prod_{i=1}^{k_1} e\left(P, sk_{ij} H\left(C_{ij} \| \text{ID}_{ij} \| \text{TS}\right)\right) \\ &= \prod_{i=1}^{k_1} e\left(pk_{ij}, H\left(C_{ij} \| \text{ID}_{ij} \| \text{TS}\right)\right). \end{aligned}$$

Note that the batch verification reduces the number of pairing operations from $2k_1$ to $k_1 + 1$. After checking the validity, FN_j aggregates all the ciphertexts and sends the aggregated message to the CC. The following steps are performed.

Step-1: FN_j aggregates the k_1 encrypted ciphertexts as

$$C_j = \prod_{i=1}^{k_1} C_{ij} \bmod n^2$$

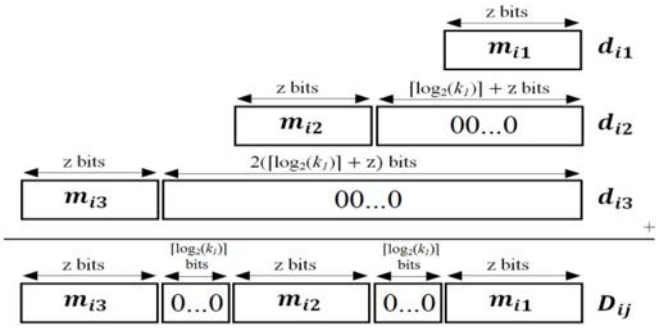

Fig. 2. Example of D_{ij} with $l = 3$.

$$\begin{aligned} &= \prod_{i=1}^{k_1} g^{\overline{D_{ij}}} \cdot r_{ij}^n \bmod n^2 \\ &= g^{\sum_{i=1}^{k_1} \overline{D_{ij}}} \cdot \left(\prod_{i=1}^{k_1} r_{ij}\right)^n \bmod n^2 \\ &= g^{\sum_{i=1}^{k_1} (D_{ij} + x_{ij}) \bmod n} \cdot \left(\prod_{i=1}^{k_1} r_{ij}\right)^n \bmod n^2. \end{aligned} \tag{8}$$

Step-2: FN_j uses its private key sk_j to compute the signature as follows:

$$\sigma_j = sk_j H\left(C_j \| \text{ID}_j \| \text{TS}\right) \tag{9}$$

where TS is the current timestamp.

Step-3: FN_j sends, to the CC, an all-inclusive report that contains $\{C_j, \text{ID}_j, \text{TS}, \sigma_j\}$.

2) Fault-Tolerant Data Aggregation: If some SMs breakdown, FN_j will not receive the corresponding packets. Let U_{ij} be the set of all legitimate SM devices and U'_{ij} be the set of failed SM devices ($U'_{ij} \in U_{ij}$). Consequently, we have

$$x_{0j} \neq \sum_{i \in U_{ij}/U'_{ij}} x_{ij} \bmod n. \tag{10}$$

Thus, this phenomenon will directly affect the correctness of the final decryption result. FN_j needs to send the set U'_{ij} to the CC. Therefore, in this case, FN_j first validates the received signatures, i.e., check whether

$$e\left(P, \sum_{i \in U_{ij}/U'_{ij}} \sigma_{ij}\right) = \prod_{i \in U_{ij}/U'_{ij}} e\left(pk_{ij}, H\left(C_{ij} \| \text{ID}_{ij} \| \text{TS}\right)\right). \tag{11}$$

If the equation does hold, it means the signatures are valid. Then, it performs aggregation on the received ciphertexts and computes the fault-tolerant aggregated ciphertext. The following specific steps are performed.

Step-1: FN_j aggregates the received ciphertexts as

$$\begin{aligned} C'_j &= \prod_{i \in U_{ij}/U'_{ij}} C_{ij} \bmod n^2 \\ &= g^{\sum_{i \in U_{ij}/U'_{ij}} (D_{ij} + x_{ij}) \bmod n} \cdot \left(\prod_{i \in U_{ij}/U'_{ij}} r_{ij}\right)^n \bmod n^2. \end{aligned} \tag{12}$$

Step-2: FN_j uses its private key sk_j to compute the signature as follows:

$$\sigma_j' = sk_j H\Big(C_j' \| \mathrm{ID}_j \| \mathrm{TS}\Big) \tag{13}$$

where TS is the current timestamp.

Step-3: FN_j sends, to the CC, a fault-tolerant report that contains $\{C_j', \mathrm{ID}_j, \mathrm{TS}, \sigma_j', U_{ij}'\}$.

E. Data Reading

This part consists of two possible phases. If the CC receives an all-inclusive report from FN_j, it will perform an *all-inclusive data reading*; otherwise, it will perform a *fault-tolerant data reading*.

1) All-Inclusive Data Reading: Upon receiving the all-inclusive report from FN_j, CC first verifies the signature according to the following equation:

$$e\big(P, \sigma_j\big) = e\big(pk_j, H\big(C_j \| \mathrm{ID}_j \| \mathrm{TS}\big)\big). \tag{14}$$

If the equation does hold, it means the signatures are valid. After checking the validity, CC decrypts the aggregated ciphertext C_j and retrieves the aggregated data by performing the following steps.

Step-1: CC decrypts the aggregated ciphertext C_j. From (8), by taking

$$M = \sum_{i=1}^{k_1} \big(D_{ij} + x_{ij}\big) \bmod n, \text{ and } R = \prod_{i=1}^{k_1} r_{ij}.$$

The report $g^M \cdot R^n \bmod n^2$ is still a ciphertext of the Paillier Cryptosystem. Thus, CC uses the tuple (λ, μ) to recover M as

$$M = L\Big(C_j^{\lambda} \bmod n^2\Big)\mu \bmod n. \tag{15}$$

Step-2: After decrypting, CC uses x_{0j} to obtain $\sum_{i=1}^{k_1} D_{ij}$ as

$$\sum_{i=1}^{k_1} D_{ij} = M - x_{0j} \bmod n. \tag{16}$$

Step-3: CC uses the decoding function to retrieve each aggregated data $\sum_{i=1}^{k_1} d_{ik}$. CC divides the binary representation of $\sum_{i=1}^{k_1} D_{ij}$ into l blocks of bits, the length of each block is $(\lceil \log_2(k_1) \rceil + z)$. Thus, the first $(\lceil \log_2(k_1) \rceil + z)$ least significant bits correspond to the aggregation result $\sum_{i=1}^{k_1} m_{i1}$ and so on. The CC then retrieves the aggregation result of each type of data as

$$\left(\sum_{i=1}^{k_1} D_{ij}\right)_2 = \sum_{i=1}^{k_1} m_{il} \Big\| \cdots\cdots \Big\| \sum_{i=1}^{k_1} m_{i2} \Big\| \sum_{i=1}^{k_1} m_{i1}. \tag{17}$$

Due to overflow during aggregation, the required extra bits cannot be more than $\lceil \log_2(k_1) \rceil$ when k_1 numbers are added. This explains why we choose to pad $(\lceil \log_2(k_1) \rceil + z)$*$(k-1)$ zeros after m_{ik} in every d_{ik}. Fig. 3. shows the aggregated data retrieved by the CC when $k_1 = 4$ and $l = 3$, where D_{ij} was generated by SM_{ij}.

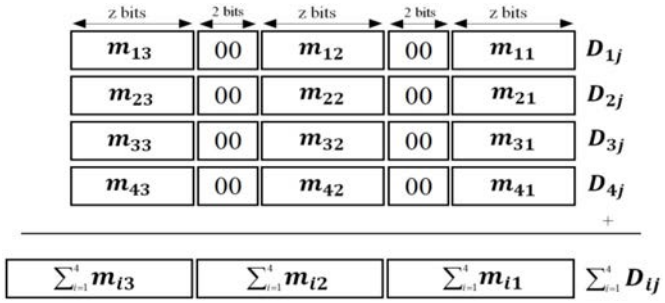

Fig. 3. Example with $k_1 = 4$ and $l = 3$.

2) Fault-Tolerant Data Reading: Upon receiving the fault-tolerant report from FN_j, CC first verifies the signature according to the following equation:

$$e\Big(P, \sigma_j'\Big) = e\Big(pk_j, H\Big(C_j' \| \mathrm{ID}_j \| \mathrm{TS}\Big)\Big). \tag{18}$$

If the equation does hold, it means the signatures are valid. CC decrypts the aggregated ciphertext C_j' and retrieves the aggregated data by performing the following steps.

Step-1: CC decrypts the aggregated ciphertext C_j' as

$$M' = L\Big(C_j'^{\lambda} \bmod n^2\Big)\mu \bmod n. \tag{19}$$

Step-2: After decrypting, CC computes x_{0j}' as

$$x_{0j}' = \sum_{i \in U_{ij}/U_{ij}'} x_{ij} \bmod n. \tag{20}$$

Step-3: CC uses x_{0j}' to obtain the fault-tolerant aggregated multidimensional data

$$\sum_{i \in U_{ij}/U_{ij}'} D_{ij}' = M' - x_{0j}' \bmod n. \tag{21}$$

Step-4: CC uses the decoding function, as described earlier, to retrieve the fault-tolerant aggregation result of each type of data as

$$\left(\sum_{i \in U_{ij}/U_{ij}'} D_{ij}'\right)_2 = \sum_{i \in U_{ij}/U_{ij}'} m_{il}' \Big\| \cdots\cdots \Big\| \sum_{i \in U_{ij}/U_{ij}'} m_{i2}' \Big\| \sum_{i \in U_{ij}/U_{ij}'} m_{i1}'. \tag{22}$$

VI. Answering Other Queries

In this section, we discuss how to adapt ESMA to answer other queries, namely, variance report query and set-aggregation report query.

A. Variance Report Query

The CC may need other statistical functions than summation such as variance to further analyze power consumption. This could allow the detection of abnormal situations and load imbalance. In the following, we present how ESMA can be adapted to achieve dual-functional MDA for FCSG, which

can support, for each type of data, aggregation of mean and variance at the same time. Observe that

$$\mathrm{VAR}(m_{i1}) = \frac{\sum_{i=1}^{k_1} m_{i1}^2}{k_1} - \left(\frac{\sum_{i=1}^{k_1} m_{i1}}{k_1}\right)^2 = \frac{1}{k_1}\sum_{i=1}^{k_1} m_{i1}^2 - \frac{1}{k_1^2}\left(\sum_{i=1}^{k_1} m_{i1}\right)^2. \quad (23)$$

If the CC needs to perform a variance analysis on multidimensional data, each SM_{ij} add the data $(m_{i1}^2, m_{i2}^2, \ldots, m_{il}^2)$ in addition to $(m_{i1}, m_{i2}, \ldots, m_{il})$ and encode them as follows:

$$d_{ik} = \left(m_{ik}^2\right)_2 \| (m_{ik})_2 \| 0^{\theta},\ k = 1, \ldots, l \quad (24)$$

where $\theta = (2\lceil \log_2(k_1)\rceil + 3z) * (k-1)$ and $(m_{ik})_2$ is padded by zeros in the left to be represented in $(\lceil \log_2(k_1)\rceil + z)$ bits.

Due to square calculation, the required bits for m_{ik}^2 cannot be more than $2z$ where z is the maximum number of bits that could represent a DT. This explains why we choose to pad $(2\lceil \log_2(k_1)\rceil + 3z) * (k-1)$ zeros in every d_{ik}. The SM_{ij} then follows the same steps as previously described to produce its report. At the end of the decryption phase, CC retrieves the queried data as follows:

$$\left(\sum_{i=1}^{k_1} D_{ij}\right)_2 = \sum_{i=1}^{k_1} m_{il}^2 \left\| \sum_{i=1}^{k_1} m_{il} \right\| \cdots\cdots \left\| \sum_{i=1}^{k_1} m_{i1}^2 \right\| \sum_{i=1}^{k_1} m_{i1}. \quad (25)$$

B. Set Aggregation Query

We suppose that there exist f subsets corresponding to f electricity consumption ranges, such that $[R_1, R_2)$, $[R_2, R_3), \ldots, [R_f, +\infty)$. The CC may need to calculate the total consumption and also the number of users of each subset in a period [21]. In the following, we show how ESMA can be adapted to achieve the data aggregation of multisubset for FCSG. We assume that if the electricity consumption data, generated by SM_{ij}, $m_i \in [R_s, R_{s+1})$, SM_{ij} will lie in the subset $\mathbb{U}_s \subset \mathbb{U}$, where $\mathbb{U} = \mathbb{U}_1 \cup \mathbb{U}_2 \cup \cdots \cup \mathbb{U}_f$, and $\mathbb{U}_s \cap \mathbb{U}_t = \emptyset$, for s, $t = 1, 2, \ldots, f$, $s \neq t$. We also have $E = E_1 + E_2 + \cdots + E_f$, E_1 is the total electricity consumption of the subset $\mathbb{U}_1$ and so on. We can then consider for each subset a block of bits in our encoding function. Each SM_{ij} encodes its data $m_i \in [R_s, R_{s+1})$ into D_{ij} as follows:

$$D_{ij} = 1 \| (m_i)_2 \| 0^{\theta},\ s = 1, \ldots, f \quad (26)$$

where $\theta = (2 * \lceil \log_2(k_1)\rceil + z) * (s-1)$ and $(m_i)_2$ is padded by zeros in the left to be represented in $(\lceil \log_2(k_1)\rceil + z)$ bits.

The SM_{ij} then follows the same steps as previously described to produce its report. Afterward, FN_j aggregates the number of users and total usage of each subset. At the end of the decryption phase, CC retrieves the queried data as follows:

$$\left(\sum_{i=1}^{k_1} D_{ij}\right)_2 = |\mathbb{U}_f| \, \|E_f\| \cdots |\mathbb{U}_2| \, \|E_2\| \, |\mathbb{U}_1| \, \|E_1. \quad (27)$$

VII. Security Analysis

In this section, we analyze the security of ESMA. In particular, we show how ESMA can protect the user's individual data, and ensure data integrity and source authentication.

A. Privacy Protection

In the SM report generation phase, the sensitive multidimensional data d_{ij} is blinded by adding the secret key x_{ij} modulo n to get the ciphertext: $C_{ij} = g^{d_{ij}+x_{ij} \bmod n} \cdot r^n \bmod n^2$. SM_{ij} sends C_{ij} to the corresponding FN_j instead of d_{ij} directly. The FN_j is then unable to decrypt the received ciphertext and to read the sensitive data d_{ij} without knowing the Paillier private key and x_{ij}. The Paillier cryptosystem is semantically secure, so the encryption is secure against any form of ciphertext analysis. User's multidimensional data $(m_{i1}, m_{i2}, \ldots, m_{il})$ in d_{ij} are also semantic secure and privacy preserving. Furthermore, due to the randomization of r, the Paillier encryption can resist dictionary attacks, since the encryption of the same data will result into different ciphertexts with very high probability. This ensures that even if the adversary $\mathcal{A}$ eavesdrops the communication or even intrudes into the FN_j database and gets C_{ij}, he will not have access to the user's private data D_{ij}.

In the data aggregation phase, FN_j performs aggregation operation directly on ciphertexts. For the CC, after decryption, it only has the data sum for every dimension. Even if $\mathcal{A}$ intrudes into the CC database, he cannot retrieve the individual multidimensional data, i.e., $(m_{i1}, m_{i2}, \ldots, m_{il})$. Therefore, the privacy of individual data is ensured in ESMA.

Furthermore, ESMA ensures that a colluding set of users cannot compromise the privacy of other users. In fact, if $\mathcal{A}$ compromises the users' privacy, he can access the private data and retrieves the secret share x_{ij}. In ESMA, there is no correlation between the randomly generated shares by TA, namely, the secret shares $x_{ij} \in \mathbb{Z}_n^*$. So, compromising the secret shares of a few users will not reveal the secret shares of remaining users. Suppose the case where $\mathcal{A}$ succeeds in compromising $k_1 - 1$ users that belong to a certain FN_j, and gets their secret shares $x_{1j}, x_{2j}, \ldots, x_{k-1j}$. For $k_1 - 1$ users, (1) can be rewritten as follows:

$$x_{0j} = \sum_{i=1}^{k_1-1} x_{ij} + x_{k_1 j} \bmod n. \quad (28)$$

Data privacy of the remaining user is still maintained because $\mathcal{A}$ does not have the sum of the secret shares x_{0j} of CC nor the Paillier private key, thereby, $\mathcal{A}$ will not be able to compromise $x_{k_1 j}$. Consequently, in ESMA, regardless of the number of compromised users, $\mathcal{A}$ cannot access the private data of the other users.

B. Data Integrity and Authentication

In ESMA, the BLS short signature [22] is adopted to sign the user's private data and the aggregated data. For the message $\{C_{ij}, \mathrm{ID}_{ij}, \mathrm{TS}, \sigma_{ij}\}$ sent by SM_{ij}, the corresponding FN_j first checks ID_{ij} and TS and then verifies the message's integrity by checking if (7) holds. We can see that each element of the

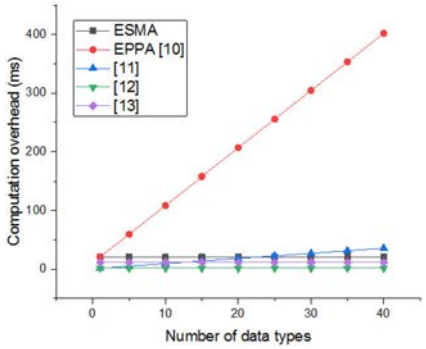

Fig. 4. Computation cost at SM.

message is involved in verification, and any manipulation of the message will cause inequality. Therefore, the message's integrity sent by SM_{ij} can be verified by FN_j. Similarly, when receiving the message $\{C_j, ID_j, TS, \sigma_j\}$ sent by FN_j, CC first checks ID_j and TS and then verifies the message's integrity by checking if (14) holds. From the equation, each element of the message is involved in verification, and any manipulation of the message will cause inequality. Therefore, the message's integrity sent by FN_j can be verified by CC.

The signature σ_{ij} of SM_{ij} is generated with the corresponding private key sk_{ij}, and the signature σ_j of FN_j is generated with the corresponding private key sk_j. Since the adversary does not know the private keys sk_{ij} and sk_j, it cannot produce the correct messages. Therefore, ESMA can guarantee that all users and their private data are legitimate.

Data integrity and authentication are then provided since the BLS signature is provably secure [22]. Consequently, $\mathcal{A}$ cannot launch active attacks without being detected. The TS included in the packet can prevent replay attacks. Any data modification or false data injection can be detected when the recipient fails to check the signature validity.

VIII. Performance Evaluation

Tonyali *et al.* [23] analyzed the cost generated when using HE in SG in terms of bandwidth and end-to-end delay. The authors state that computation and communication costs and scalability impact severely the network reliability and must be considered. Consequently, in this section, we first compare, in terms of computation and communication cost, ESMA with previous works, and then we analyze its scalability.

A. Computation Cost

Using well-known cryptographic libraries, MIRACL [24], and PBC [25], we compute the cost of the cryptographic operations used in ESMA and previous schemes. We consider for Paillier encryption a 1024-b n and for pairing a base field size of 160 b. For ECC, we use the standard curve secp160-r1. Table III shows the results obtained on a computer with Intel Core i5-2430 2.4-GHz CPU and 2-GB RAM. Note that multiplication in $\mathbb{Z}_{n^2}$, modular addition, and hash operations are negligible compared to exponentiation in $\mathbb{Z}_{n^2}$ and pairing operations. We compare ESMA with the schemes that achieve MDA, namely, EPPA [10], Boudia *et al.* [11], Ming *et al.* [12], and Zuo *et al.* [13] (see Table I).

TABLE III
Cost of Cryptographic Operations

Symbol	Operation	Time (ms)
C_e	Exponentiation in $\mathbb{Z}_{n^2}$	9,78
C_m	Multiplication in $\mathbb{G}_1$	1,18
C_p	Pairing	22,84
C_{pm}	ECC Point Multiplication	0,44
$C_{E\text{-}ElGamal}$	El Gamal Encryption	5,81
$C_{D\text{-}ElGamal}$	El Gamal Decryption	3,14
C_{exp}	Exponentiation in $\mathbb{Z}_n$	2,88

In ESMA, when SM_{ij} produces its report, it requires 2 C_e to generate C_{ij} and C_m to generate σ_{ij}. In EPPA [10], SM_{ij} performs $(l+1)C_e$ and C_m to generate the ciphertext and the signature. In [11], SM_{ij} performs $2(l+1)C_{pm}$. In [12], SM_{ij} performs 4 C_{pm}. In [13], SM_{ij} performs $C_{E-ElGamal}$ and $2C_{\text{exp}}$. We present a comparison in Fig. 4. The results show that ESMA significantly reduces the computation cost at the user side in comparison with EPPA scheme which also utilizes Paillier to achieve MDA. In [11], the cost increases linearly with the number of DTs. In [12] and [13], the cost is efficient; however, as we will see later, this advantage is lost at CC where the decryption operation requires solving DLP which is a heavy computation in the case of large plaintexts.

In ESMA, EPPA, and [13], when FN_j (gateway in the case of EPPA and [13]) receives all of the reports from the covered SM_{ij}, it first performs a batch verification to verify the data that needs $(k_1+1)C_p$. Then, it aggregates the ciphertexts and produces a signature, which consists of several negligible multiplications in $\mathbb{Z}_{n^2}$ and C_m (multiplications in $\mathbb{Z}_n$ and C_{exp} in [13]). Consequently, at FN_j, the costs are almost the same, except for [11] and [12], which employ an Elliptic curve-based batch verification that is more efficient than pairing one. However, in the fog computing paradigm, FNs are considered to have more computational abilities than traditional gateways. Thus, the aggregation operation can be efficiently computed.

In ESMA and EPPA [10], when CC (OA in the case of EPPA) receives a packet from FN_j, it first performs $2C_p$ to verify the data and then retrieves the aggregation result by performing C_e. In [11], the CC performs $3C_{pm}$ and l times a reverse mapping function which requires solving the ECDLP to recover M. To verify and retrieve the data in [12], the CC performs $4C_{pm}$ and a computation of the ECDLP to recover M, while in [13], the CC performs 2 (k_1+1) C_p, $C_{\text{D-ElGamal}}$ and a computation of the DLP to recover M. By considering Pollard's lambda method to compute DLP and ECDLP, we conduct experiments to compute the execution time at CC using MIRACL. The method requires a time complexity of $O(l \cdot \sqrt{k_1 \cdot 2^z})$, $O(\sqrt{l \cdot k_1 \cdot 2^z})$, $O(\sqrt{(l+1) \cdot k_1 \cdot 2^z})$ in [11]–[13], respectively. The results presented in Fig. 5 show that regardless of the size of the aggregation result M, EPPA and ESMA have the same computation overhead at CC. However, in the other works, the higher is the size of the aggregation result, the higher is the computation time. In fact, an aggregate size higher than 40 b cannot be efficiently recovered at CC at all. Ugus *et al.* [26] stated that in the El Gamal additive encryption scheme, the final aggregation must be small enough for realistic applications (e.g., 3 bytes) allowing

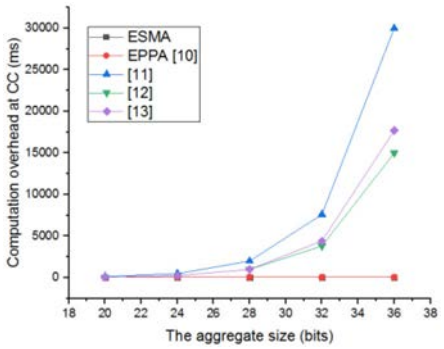

Fig. 5. Computation cost at CC vs. The aggregate size.

a successful recovery at the decryption device. Consequently, the schemes of Boudia *et al.* [11], Ming *et al.* [12], and Zuo *et al.* [13] cannot be practical for large aggregation results for SG. The super increasing sequence has been utilized in [12] and [13] with El Gamal HE to achieve MDA. In these two schemes, the CC has to solve the DLP to retrieve the aggregated data. Boudia *et al.* [11] used El Gamal HE with multiple public keys. However, in this case, also, the CC has to compute the ECDLP to retrieve the aggregated data. In the aforementioned schemes, the authors use heavy computation either to encrypt [10], [11] or to decrypt [11]–[13] the multidimensional data. Consequently, ESMA is the most efficient.

In 2019, Saleem *et al.* [18] have proposed a fault-tolerant mechanism to achieve resilience to node failures in their ODA scheme for FCSG. However, in their proposal, the fault-tolerant decryption cost increases at CC with the increasing number of faulty SMs. In fact, the more are the number of faulty SMs more cryptographic operations are executed at CC. Furthermore, the decryption at CC requires solving DLP to recover the aggregation result. In ESMA, regardless of the number of faulty SMs, the cost of the fault-tolerant decryption at CC is always the same. Fig. 6 shows the efficiency of our fault-tolerant decryption compared to that of Saleem *et al.* [18].

B. Communication Cost

In MDA schemes, l types of data are encrypted and sent by each SM to the corresponding aggregator. The communication overhead can be divided into two parts: 1) SM-to-FN communication and 2) FN-to-CC communication. In ESMA, C_{ij}, ID_{ij}, TS, and σ_{ij} are sent from SM to FN_j, where $C_{ij} \in \mathbb{Z}_{n^2}$ and $\sigma_{ij} \in \mathbb{G}_1$. Let the size of ID_{ij} and TS be all 8 B. Therefore, the communication cost SM-to-FN is 2048 + 64 + 160 = 2272 b. Next, we analyze FN-to-CC communication. In ESMA, $\{C_j, \mathrm{ID}_j, \mathrm{TS}, \text{ and } \sigma_j\}$ are sent from FN_j to CC, where $C_{ij} \in \mathbb{Z}_n^2$ and $\sigma_{ij} \in \mathbb{G}_1$. Therefore, the communication cost FN-to-CC is $2048 + 64 + 160 = 2272$ b. The comparison of communication cost with previous MDA schemes is shown in Table IV.

We depict in Fig. 7 the comparison in terms of communication cost SM-to-FN for 20 types of data. It can be seen that ESMA incurs lower communication overhead than [11].

TABLE IV
COMMUNICATION OVERHEAD COMPARISON

Scheme	SM-to-FN	FN-to-CC
Lu et al. [10]	2308 bits	2308 bits
Merad et al. [11]	160 l +544 bits	160 l +544 bits
Ming et al. [12]	704 bits	704 bits
Zuo et al. [13]	2112 bits	2112 bits
ESMA	2272 bits	2272 bits

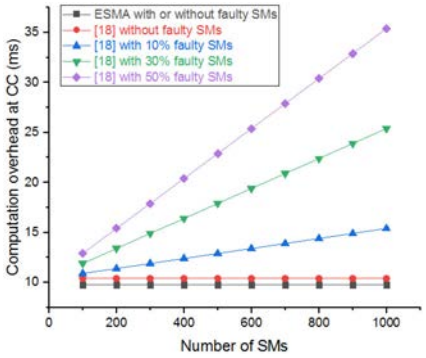

Fig. 6. FT decryption cost at CC.

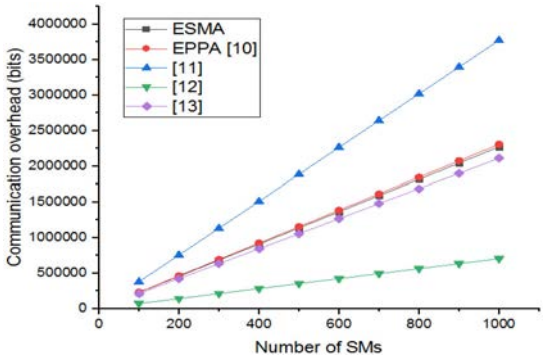

Fig. 7. SM-to-FN communication cost.

The cost of communication is approximately equal to that of [10] and [13]. However, in comparison with [9], the communication overhead is higher in ESMA. The reason is that Ming *et al.* [12] employed in their scheme elliptic curve groups for encryption and signature, which leads to cost-efficient communications. Nevertheless, first, as previously mentioned, the main drawback of their proposal is that the CC has to compute the ECDLP, which severely limits the upper bound. Second, they consider a finite field with a 160-b q, the plaintext space is then severely limited for the considered super-increasing sequence, which has an impact on the number of DTs as well. The same observation can be made for [11] and [13]. In ESMA, there is no extra computation at SM and CC. The plaintext is up to 1024 b and an important number of DTs can be supported in a single ciphertext.

C. Scalability Analysis

In ESMA, we use an encoding function to structure the multidimensional data in $\mathbb{Z}_n$. In our implementation, we consider a 1024-b n in Paillier encryption. However, Paillier can support larger key sizes, such as 2048 and 4096 b for improved

TABLE V
SCALABILITY ANALYSIS

$\|n\|$	z	k_1	l
1024	16 bits	125	44
		250	42
		500	40
		1000	39
	32 bits	125	26
		250	25
		500	24
		1000	24

TABLE VI
STORAGE OVERHEAD COMPARISON

Scheme	SM	FN
Lu et al. [10]	1024 (l +1) +192 bits	320 k_l + 192 bits
Merad et al. [11]	320 (l +1) + 192 bits	320 (k_l+1) + 192 bits
Ming et al. [12]	992 + l $\|a_i\|$ bits	320 (k_l+1) + 192 bits
Zuo et al. [13]	1568 + l $\|a_i\|$ bits	512 k_l + 192 bits
ESMA	3264 bits	320 k_l + 192 bits

security. Recall that each SM_{ij} encodes its l types of data $(m_{i1}, m_{i2}, \ldots, m_{il})$ into $(d_{i1}, d_{i2}, \ldots, d_{il})$ as

$$d_{ik} = (m_{ik})_2 \| 0^{\theta}, \ k = 1, \ldots, l$$
$$\text{where } \theta = \left(\left\lceil \log_2(k_1) \right\rceil + z\right) * (k-1).$$

In Table V, we show, according to n, z, and k_1, the number of DTs l that can be supported in a single ciphertext. The table shows that using our encoding function, an important number of DTs l can be supported in a single Paillier ciphertext. For instance, with $z = 16$ b and 500 reporting SMs, 40 DTs can be supported in a single ciphertext. Note that the number z can be significantly reduced with a reference technique, such as presented in [27] for efficient communications in sensor networks [28]. The plaintext space in previous works cannot support a large number of DT as mentioned earlier.

D. Storage Analysis

In ESMA, each SM stores in its memory its ID, the public key of CC, a private key, and a secret share. Each FN stores its ID, the public key of each covered SM, and a private key. In EPPA [10], each SM stores its ID, l generators, n, and a private key. Each FN stores the public key of each covered SM and a private key. In [11], each SM stores its ID, l public keys, a generator G, and a private key. Each FN stores the public key of each covered SM and a private key. In [12], each SM stores its ID, l big primes a_i, the CC's public keys, a generator P, a secret share, and a private key. Each FN stores the public key of each covered SM, P, and a private key. In [13], each SM stores its ID, the CC's public key, l big primes a_i, g, and a private key. Each FN stores the public key of each covered SM and a private key. A comparison with previous works is presented in Table VI. Note that the points on the curve are considered in their uncompressed form. The main observation we can make is that the storage overhead in previous schemes depends on the number of DTs l. In our work, the overhead is 3264 b whatever the number l.

IX. CONCLUSION

In this article, we proposed ESMA, an efficient and secure MDA for FCSG. ESMA can structure and encrypt the multidimensional data into a single Paillier ciphertext. Thanks to the encoding function employed, which also gives the ability to the CC to recover the aggregated data for every dimension efficiently and securely. The security analysis shows that the privacy, confidentiality, integrity, and authentication of the data are provided. The performance analysis shows the scalability advantage of ESMA, the efficiency of our fault-tolerant mechanism, and also, the cost-efficiency of ESMA in terms of computation and communication. ESMA can also be adapted to respond to other queries than summation. Therefore, it satisfies the application needs of SG for smart cities.

REFERENCES

[1] G. Dileep, "A survey on smart grid technologies and applications," *Renew. Energy*, vol. 146, pp. 2589–2625, Feb. 2020.

[2] S. Kaneriya *et al.*, "Data consumption-aware load forecasting scheme for smart grid systems," in *Proc. IEEE Globecom Workshops*, 2018, pp. 1–6.

[3] S. Tanwar, S. Kaneriya, N. Kumar, and S. Zeadally, "ElectroBlocks: A blockchain-based energy trading scheme for smart grid systems," *Int. J. Commun. Syst.*, vol. 33, no. 15, 2020, Art. no. e4547.

[4] A. Kumari, D. Vekaria, R. Gupta, and S. Tanwar, "Redills: Deep learning-based secure data analytic framework for smart grid systems," in *Proc. IEEE ICC Workshops*, 2020, pp. 1–6.

[5] M. Shateri, F. Messina, P. Piantanida, and F. Labeau, "Real-time privacy-preserving data release for smart meters," *IEEE Trans. Smart Grid*, vol. 11, no. 6, pp. 5174–5183, Nov. 2020.

[6] F. Li, B. Luo, and P. Liu, "Secure information aggregation for smart grids using homomorphic encryption," in *Proc. IEEE SmartGridComm*, 2010, pp. 327–332.

[7] A. Kumari, S. Tanwar, S. Tyagi, N. Kumar, M. S. Obaidat, and J. J. P. C. Rodrigues, "Fog computing for smart grid systems in the 5G environment: Challenges and solutions," *IEEE Wireless Commun.*, vol. 26, no. 3, pp. 47–53, Jun. 2019.

[8] M. S. H. Nazmudeen, A. T. Wan, and S. M. Buhari, "Improved throughput for power line communication (PLC) for smart meters using fog computing based data aggregation approach," in *Proc. IEEE Int. Smart Cities Conf.*, 2016, pp. 1–4.

[9] 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.

[10] 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.

[11] O. R. M. Boudia, S. M. Senouci, and M. Feham, "Elliptic curve-based secure multidimensional aggregation for smart grid communications," *IEEE Sensors J.*, vol. 17, no. 23, pp. 7750–7757, Dec. 2017.

[12] Y. Ming, X. Zhang, and X. Shen, "Efficient privacy-preserving multidimensional data aggregation scheme in smart grid," *IEEE Access*, vol. 7, pp. 32907–32921, 2019.

[13] X. Zuo, L. Li, H. Peng, S. Luo, and Y. Yang, "Privacy-preserving multidimensional data aggregation scheme without trusted authority in smart grid," *IEEE Syst. J.*, early access, Jun. 16, 2020, doi: 10.1109/JSYST.2020.2994363.

[14] M. Badra and S. Zeadally, "Lightweight and efficient privacy-preserving data aggregation approach for the smart grid," *Ad Hoc Netw.*, vol. 64, pp. 32–40, Sep. 2017.

[15] 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.

[16] J.-N. Liu, J. Weng, A. Yang, Y. Chen, and X. Lin, "Enabling efficient and privacy-preserving aggregation communication and function query for fog computing based smart grid," *IEEE Trans. Smart Grid*, vol. 11, no. 1, pp. 247–257, Jan. 2020.

[17] F. Y. Okay, S. Ozdemir, and Y. Xiao, "Fog computing-based privacy preserving data aggregation protocols," *Trans. Emerg. Telecommun. Technol.*, vol. 31, no. 4, 2020, Art. no. e3900.
[18] A. Saleem *et al.*, "FESDA: Fog-enabled secure data aggregation in smart grid IoT network," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6132–6142, Jul. 2020.
[19] S. Zhao *et al.*, "Smart and practical privacy-preserving data aggregation for fog-based smart grids," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 521–536, Aug. 2020.
[20] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptograph. Technol.*, 1999, pp. 223–238.
[21] S. Li, K. Xue, Q. Yang, and P. Hong, "PPMA: Privacy-preserving multisubset data aggregation in smart grid," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 462–471, Feb. 2018.
[22] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proc. Int. Conf. Theory Appl. Cryptograph. Technol.*, 2003, pp. 416–432.
[23] S. Tonyali, R. Munoz, K. Akkaya, and U. Ozgur, "A realistic performance evaluation of privacy-preserving protocols for smart grid AMI networks," *J. Netw. Comput. Appl.*, vol. 119, pp. 24–41, Oct. 2018.
[24] Certivox. (2014). *Multiprecision Integer and Rational Arithmetic C/C++ Library (MIRACL)*. [Online]. Available: https://github.com/miracl/MIRACL
[25] B. Lynn. *PBC Library*. Accessed: Jun. 14, 2013. [Online]. Available: https://crypto.stanford.edu/pbc/
[26] O. Ugus *et al.*, "Optimized implementation of elliptic curve-based additive homomorphic encryption for wireless sensor networks," 2009. [Online]. Available: arXiv:0903.3900.
[27] H. Sanli, S. Ozdemir, and H. Cam, "SRDA: Secure reference-based data aggregation protocol for wireless sensor networks," in *Proc. IEEE 60th Int. Conf. Veh. Technol. (VTC-Fall)*, vol. 7, Sep. 2004, pp. 4650–4654.
[28] H. Fouchal, Y. Francillette, P. Hunel, and N. Vidot, "A distributed power management optimisation in wireless sensors networks," in *Proc. LCN*, 2009, pp. 763–769.