LPDA-EC: A Lightweight Privacy-Preserving Data Aggregation Scheme for Edge Computing

Jiale Zhang[†], Yanchao Zhao[†‡], Jie Wu[§] and Bing Chen[†‡]
[†]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[‡]Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China
[§]Center for Networked Computing, Temple University, Philadelphia, USA
Email:{jlzhang, yczhao, cb_china}@nuaa.edu.cn; jiewu@temple.edu

***Abstract*—Edge computing has emerged as the key enabling technology that empowers the IoT with intelligence and efficiency. In this data enriched infrastructure, privacy-preserving data aggregation (PPDA) is one of the most critical services. However, the security and privacy-preserving requirements and online computational cost still present practical concerns in edge computing for resource-constraint edge terminals. To cope with this challenge, we present a lightweight privacy-preserving data aggregation scheme named LPDA-EC for edge computing system by employing the online/offline signature technique, Paillier homomorphic cryptosystem, and double trapdoor Chameleon hash function in this paper. The proposed LPDA-EC scheme can achieve data confidentiality and privacy-preserving, ensuring that the edge server and control center are agnostic of the user's private information during the whole aggregation process. Through detailed analysis, we demonstrate that our scheme is existentially unforgeable under chosen message attack (EU-CMA) and ensures data integrity with formal proofs under q-Strong Diffie-Hellman (q-SDH) assumptions. Numerical results indicate that the LPDA-EC scheme has less computational and communication overheads.**

***Index Terms*—Edge computing, Privacy-preserving, Data aggregation, Homomorphic cryptosystem, Chameleon hash function, Online/offline signature.**

I. INTRODUCTION

A. Background

With the explosive growth of Internet of Things (IoT) devices and wide deployment of IoT infrastructure, all IoT-based typical applications, such as smart grid [1], smart healthcare [2], smart city [3], and vehicular sensing system [4], are interconnected via a network and operate on a number of IoT devices that frequently collect and transmit data to the cloud center for observing the real-time and intelligent decisions. For example, in the smart grid application system, data reports generated from distributed smart meters are transmitted to the remote control center via the Internet for further analysis, and the control center can monitor the power delivery and electricity consumption information periodically to make real-time decisions. Note that these IoT-based smart applications generate massive volumes of data and transfer the data to the remote cloud center for big data analytics. In this situation, the traditional IoT data processing architecture has come to the bottleneck and cannot handle the IoT big data transmission and processing due to the bandwidth limitation and resources constraint [5].

Edge computing [6] is a promising distributed model that allows storing and processing data at the edge of the network with the edge server, which will not only reduce the transmission overhead but also improve the real-time processing capability. Through the combination of edge computing and cloud computing, the real-time data can be collected and aggregated by the edge server and then forwarded to the cloud computing for further analysis, as shown in Fig. 1, thereby overcoming the shortcomings of traditional IoT architecture such as bandwidth limitation and resources constraint [7]. However, the security and privacy issues still present practical concerns for edge computing, since the edge server deployed at the network edge cannot be fully trusted. For example, in order to obtain services and benefits, users need to share their collected data with the edge server, and these sensed data (e.g., electricity consumption in the smart grid) may contain users' private information, that may be eavesdropped upon untrusted edge servers [8]. Thus, the idea of privacy-preserving data aggregation (PPDA) transmission has emerged to solve the privacy leakage problem in IoT-based application scenarios, and many PPDA schemes [9–15] have been proposed. However, most of them are not suitable for the edge computing system due to the high computational costs and the frequent data transmission. Therefore, we present a lightweight data aggregation scheme for edge computing in this paper that can simultaneously achieve privacy-preserving and lightweight aggregation.

B. Related Work and Motivations

Due to the frequency of data transmission and the importance of personal privacy, many data aggregation schemes have been proposed recently. Li et al. [9] proposed an in-network incremental data aggregation scheme by using the Paillier additive homomorphic cryptosystem. The data can be aggregated following a network topology-based aggregation tree. Lu et al. [10] presented an efficient and privacy-preserving aggregation scheme named EPPA for smart grids, which utilized the extended Paillier cryptosystem to achieve secure data aggregation. This scheme also exploits the super-increasing sequence to structure multidimensional data into one dimensional, which can reduce the communication overhead. Later, Li et al. [11] presented EPPDR by combining homomorphic encryption and key evolution technique, which supports the adaptive private

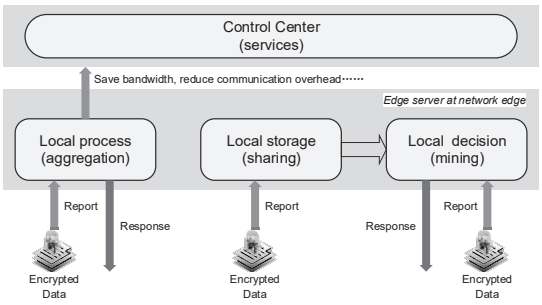

Fig. 1. Edge computing enhanced privacy-preserving data aggregation

key evolution and forward secrecy of users' session keys. In the same year, Fan et al. [12] proposed the first privacy-enhanced aggregation scheme named PEDA against internal attackers by injecting blinding factors in the report generation phase, which can resist the formidable attackers. In 2015, Ni et al. [13] presented a security-enhanced data aggregation scheme for smart grid communications based on homomorphic encryption, homomorphic authenticators, and trapdoor hash function, which can achieve data confidentiality and integrity against malicious aggregator during the aggregation process. Later, the authors in [13] further designed an efficient data aggregation scheme [14] to resist privacy exposure without any trusted third party by using the random noisy technique. Recently, Lu et al. [15] designed an efficient data aggregation scheme for fog-enhanced IoT applications to aggregate hybrid data into one, while can early filter false data at the fog nodes.

Notice that the schemes above all consider the aggregation scheme to protect data privacy and reduce communication overhead simultaneously, while the computational complexity is still an urgent problem to be solved during the frequent aggregation requests in edge computing system. Specifically, in a certain aggregation scheme, the time-consuming operations (e.g., paring and exponentiation operation) are mainly concentrated on the signature and verification processes due to the data integrity requirement. Therefore, there is a critical need to design a lightweight privacy-preserving data aggregation framework for complicated edge computing system.

C. Our Work

In this paper, we design a Lightweight Privacy-preserving Data Aggregation scheme for Edge Computing system (**LPDA-EC**) which can achieve data confidentiality, privacy-preserving, and lightweight aggregation simultaneously to address the above challenges. Specifically, our contributions can be summarized in the following three aspects:

- *Lightweight Aggregation:* In our LPDA-EC scheme, the time-consuming online signature computational cost is transferred to the offline phase by employing the online/offline signature technique and double trapdoor Chameleon hash function. The users only need a small number of operations for the online computation.
- *Privacy-Preserving:* We give the detailed analysis to show that our proposed LPDA-EC scheme can achieve confidentiality and privacy-preserving under our defined security model.
- *Unforgeability Signature:* The online/offline signature in our LPDA-EC scheme is proved existentially unforgeable under chosen message attacks.

II. PRELIMINARIES

In this section, we briefly review the bilinear pairing technique [16], Paillier Cryptosystem [17], online/offline signatures [18][19], and security definitions [20], which will facilitate the understanding of our LPDA-EC scheme.

A. Bilinear Pairing Setting

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p, and g be a generator of $\mathbb{G}$. Consider a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ satisfies the following properties [16]:

- *Bilinear:* For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p^*$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- *Nondegenerate:* g should satisfy $e(g, g) \neq 1_{\mathbb{G}_T}$.
- *Computable:* $e(u, v)$ should be computable.

Definition 1. q-Strong Diffie-Hellman Problem (q-SDH) [18]: Solving the q-SDH problem in $\mathbb{G}$ is to compute a pair (m, σ_x) where $(m, x) \in \mathbb{Z}_p^*$, given a $(q+1)$-tuple $(g, g^x, g^{(x^2)}, ..., g^{(x^q)})$. We say that the q-SDH problem is (q, t, ϵ)-hard to solve, for any t-time adversary $\mathcal{A}$, the following probability is negligible in ϵ.

$$Pr[\mathcal{A}(g, g^x, g^{(x^2)}, ..., g^{(x^q)}) = (m, \sigma_x), m \in \mathbb{Z}_p^*] < \epsilon. \quad (1)$$

Theorem 1. *We say that the (q, t, ϵ)-SDH assumption holds in $\mathbb{G}$ if no t-time algorithm has advantages at least ϵ in solving the q-SDH problem in $\mathbb{G}$.*

B. Paillier Homomorphic Cryptosystem

For concreteness and without loss of generality, our LPDA-EC scheme is based on the Paillier cryptosystem. The concrete description of Paillier cryptosystem is shown as follows:

- *KeyGen:* Given two large primes (p, q), the RSA modulus $n = pq$ and the Carmichael function $\lambda = (p-1)(q-1)$ are computed. g is a generator of $\mathbb{Z}_{n^2}^*$ with an order n, meaning that $g^n \mod n^2 = 1$. Define a function $L(u) = \frac{u-1}{n}$ and further calculate $\mu = \left(L(g^\lambda \mod n^2)\right)^{-1}$. The public key is $pk = (n, g)$ and the corresponding private key is $sk = (\lambda, \mu)$.
- *ENC:* Given a plaintext message $m \in \mathbb{Z}_n$ and the random number r is chosen such that $gcd(r, n) = 1$. The ciphertext can be computed as $c = g^m \cdot r^n \mod n^2$.
- *DEC:* Given the ciphertext $c \in \mathbb{Z}_{n^2}^*$, the corresponding plaintext message can be recovered as $m = L(c^\lambda \mod n^2)\mu \mod n$.

The Paillier homomorphic cryptosystem can be proved to be semantically secure against chosen plaintext attack based on decisional composite residuosity problem, and the correctness and security proof can be found in [17].

C. Online/Offline Signatures

An online/offline signature scheme can split a signing algorithm into two phases. The first phase is performed in the *offline* phase before a message to be signed is presented and the highest complexity operations are accomplished in this phase. The second phase is performed in the *online* phase after the massage is given. It is very lightweight and can be calculated easily by a resource-constraint end device. Besides message signature, the verification of signature can be also separated into offline and online phases by using the Double Trapdoor Chameleon Hash (DTCH) function [21]. In our edge computing system, the offline phase of signature and verification can be executed as a background computation in edge server.

The DTCH function is a very useful method to construct an online/offline signature scheme, which can achieve the fully adaptively secure one-time signature property. The DTCH function used in our work can be described as follows: Let $\mathbb{G}$ be a group generated by prime order p_1, and let $g_1 \in \mathbb{G}$ be a generator. Choose two random elements (trapdoor keys) y, z from $\mathbb{Z}_{p_1}^*$ and compute $g_2 = g_1^y, g_3 = g_1^z$. The public key is $pk = (g_1, g_2, g_3)$ and the corresponding private key is $sk = (y, z)$. For the given input elements of chameleon hash (r, s, u) from $\mathbb{Z}_p$, the output is a hash value of $\mathbb{G}$, which can be defined as $H_{ch}(r, s, u) = g_1^r \cdot g_2^s \cdot g_3^u$.

D. Security Definitions

Definition 2. Unforgebility: For an online/offline signature scheme, the existential unforgeability under chosen message attacks (EU-CMA) is defined in the following game [20]. This game is carried out between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. The adversary is allowed to make queries to an offline signing oracle $sig^{off}(sk)$ and an online signing oracle $sig^{on}(sk, St_i, m_i)$ where st_i means the state information of singer. We assume that the adversary $\mathcal{A}$ is able to make the t-th online signature query after the i-th offline signature query has been made, which is reasonable since the signer always executes his i-th offline signing before his i-th online signing. The advantage in existentially forging a signature of the adversary $\mathcal{A}$ is:

$$Adv_{\mathcal{A}} = Pr\left[\begin{matrix} Ver^{on}(pk, m^*, \sigma^*) = 1 : (pk, sk) \leftarrow \\ KeyGen(1^k); (m^*, \sigma^*) \leftarrow \mathcal{A}^{(\sigma^{off}, \sigma^{on})} \end{matrix}\right]. \quad (2)$$

III. MODELS AND DESIGN GOALS

In this section, we formalize the system model, security requirements, and identify our design goals.

A. System Model

In our system model, we formalize the communications among all entities as depicted in Fig. 2. Specifically, there are four entities that include a trusted authority, a control center, an edge server, and edge terminals in the system model of the proposed scheme.

- *Trusted Authority (TA):* The TA is a fully trusted third party whose duty is to bootstrap the whole system and distribute the key materials. We assume that there are secure channels between the TA and other entities, which support the transmission of these key materials. In general, after bootstrapping the system, the TA will not be involved in the subsequent process.
- *Control Center (CC):* The CC's duty is to collect all users' data from the edge server and make some analytics according to the realistic requirements.
- *Edge Server (ES):* ES is a core entity for edge computing system with certain computation capability, which is deployed at the edge of the network and serves as a relay and aggregator role between the CC and edge terminals.
- *Edge Terminal (ET):* ET represent a set of devices owned by users. Each terminal ET_i is equipped with sensing and communication module, which enables ET_i to collect the private data m_i and transfer its report P_i to the control center via ES.

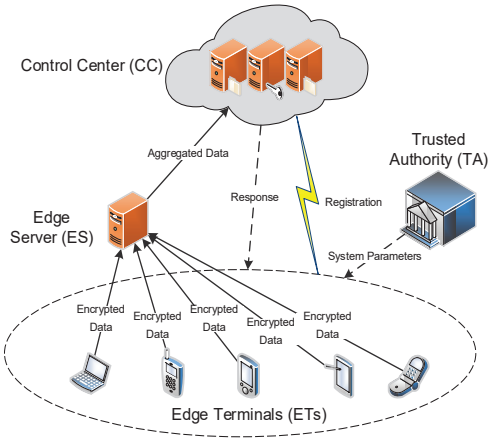

Fig. 2. Architecture of our proposed data aggregation scheme

Since ET's computational resources are usually constrained, the security algorithms with high computational complexity (or time-consuming) cannot be deployed. The shortcoming of ET motivates us to design a lightweight security mechanism for edge computing system.

B. Security Requirements

In our security model, we assume that the TA and CC are fully trusted, while ES is *honest-but-curious*. On the one hand, they faithfully follow the designated aggregation protocol. On the other hand, they are curious and attempt to disclose users' sensitive information. In addition, there exists an adversary $\mathcal{A}$ residing in edge computing communication channels to intercept the transmission of reports from aggregator and users. The adversary $\mathcal{A}$ could also launch some activity attacks or intrude the internal database to threaten the data integrity and privacy. Therefore, to ensure the safe transmission of reports and preserve the privacy of users, the following security requirements should be satisfied.

- *Confidentiality:* Confidentiality is a fundamental requirement that prevents the unauthorized parties from accessing the users' private data even if this adversary can eavesdrop the communication channels.

TABLE I
THE DETAILED DESCRIPTION OF REGISTRATION PHASE

Edge Server		**Edge Terminal**		**Control Center**
		• Choose $X_i \in \mathbb{Z}_{p_1}^*, ID_i, TS_i$ • Calculate $(Sig_{sk}, Ver_{pk}) = (X_i, Y_i = g_1^{X_i})$ $\alpha_i = g_1^{H_1(ID_i \| \| TS_i \| \| k_i)} = g_1^{r_i}$ $\beta_i = r_i - X_i H_2(\alpha_i)$	$\xrightarrow{Y_i, \alpha_i, \beta_i}$	• Verify $\alpha_i = g_1^{\beta_i} Y_i^{H_2(\alpha_i)}$ • Publish (Y_i, α_i, β_i)
Registration Phase				
		• Choose $y, z, s_i, u_i \in \mathbb{Z}_{p_1}^*$ • Calculate $g_2 = g_1^y, g_3 = g_1^z$ $H_{ch_i} = g_1^{r_i} \cdot g_2^{s_i} \cdot g_3^{u_i}$		
• Maintain T_i^{off}	$\xleftarrow{T_i^{off} = (ID_i \| \| TS_i \| \| \sigma_i^{off})}$	$\sigma_i^{BLS} = \left(H_0(H_{ch_i})\right)^{X_i}$ $\sigma_i^{off} = (\sigma_i^{BLS}, H_{ch_i})$	$\xrightarrow{(g_1, g_2, g_3)}$	• Publish (g_1, g_2, g_3) *Offline Signature Generation*

- *Authentication and Integrity:* Authentication ensures the identity of a user is authorized, which is to guarantee the encrypted report is truly generated by a legal user. Then, the integrity is to prevent the encrypted reports from being modified by the adversary $\mathcal{A}$ during the transmission. Any unauthorized and modified report can be detected by the CC when reading the report.
- *Privacy-preserving:* As long as the aforementioned security requirements can be guaranteed, the private information of users including sensitive data, personal identities, and real-time location information can achieve the privacy-preserving requirement.

C. Design Goals

Our design goal is to propose a lightweight privacy-preserving data aggregation scheme for edge computing under the aforementioned system model and security requirements. Specifically, our scheme should capture the following objectives:

- *Security and Privacy:* As stated above, all security requirements (i.e, confidentiality, authentication, and integrity) should be guaranteed for our LPDA-EC scheme, that is, the CC and ES can detect the illegal operations from adversaries and the reliable reports can be received by the CC and ES in a trusted way. Meanwhile, the users' privacy should be protected as well in our proposed scheme, which means that no one can read any individual user's data and the aggregation results can only be obtained by the trusted CC.
- *Efficiency:* The proposed aggregation scheme should be efficient. This means that the computation cost at ET should be as less as possible, since the ET are resource-constrained devices. In addition, the communication-effectiveness should also be achieved in our proposed scheme to support the frequent aggregation requests in a certain period and the simultaneous transmission of large amounts of reports.

IV. PROPOSED LPDA-EC SCHEME

In this section, we present our lightweight privacy-preserving data aggregation scheme for edge computing system (LPDA-EC) by utilizing the online/offline signature and verification technique, homomorphic cryptosystem, and double trapdoor hash functions, which mainly consists of five phases: system initialization, registration, report generation, report aggregation, and report reading.

A. System Initialization

In our edge computing system, there exists a single TA who can bootstrap the whole system. Specifically, in the system initialization phase, on input the security parameters (k, k_1), TA first randomly chooses two distinct larger primes (p, q), and computes the RSA modulus $n = pq$ and the Carmichael's function $\lambda = lcm(p-1, q-1)$, where $|p| = |q| = k$. Then, TA defines a function $L(x) = \frac{\mu - 1}{n}$ where μ can be calculated as $\mu = \left(L(p^\lambda \mod n^2)\right)^{-1}$. TA also chooses a generator $g \in \mathbb{Z}_{n^2}^*$. Thus, the Paillier Cryptosystem's public key is $PK_P = (n, g)$, and the corresponding private key is $SK_P = (\mu, \lambda)$.

Then, the TA generates two multiplicative cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of the same prime order p_1, where $|p_1| = k_1$, and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The TA further chooses a generator $g_1 \in \mathbb{G}$ and three secure cryptographic hash functions $H_0 : \{0,1\}^* \to \mathbb{G}$, $H_1 : \{0,1\}^* \to \mathbb{Z}_{p_1}^*$, $H_2 : \mathbb{G} \to \mathbb{Z}_{p_1}^*$ and a Chameleon hash function $H_{ch} : \mathbb{Z}_{p_1}^* \to \mathbb{G}$. In addition, we assume that the number of ET in a certain aggregation time slot is ω.

After the above parameter settings, the TA releases the system parameters as

$$SP_{pub} = \{p_1, n, g, \mathbb{G}, \mathbb{G}_T, e, g_1, \omega, H_0, H_1, H_2, H_{ch}\}, \quad (3)$$

and the master keys will be assigned to the CC via a secure channel as

$$msk = (\lambda, \mu, p, q). \quad (4)$$

B. Registration

When a user terminal ET_i joins the edge computing system, it needs to register to the CC and then send the offline signature to ES. The whole registration and offline signature generation phase is shown in table I.

- *User Registration:* ET_i first chooses a secure signature scheme $Sig_{sk}()/Ver_{pk}()$ and generates a random value $X_i \in \mathbb{Z}_{p_1}^*$ as the signature private key. Then ET_i calculates the corresponding verification public key as

TABLE II
THE DETAILED DESCRIPTION OF REPORT GENERATION PHASE

Edge Server		**Edge Terminal**
• Check the time stamp: TS_i		
• Verify the offline signnature σ_i^{off} with: $e(g_1, \sigma_i^{BLS}) = e\big(Y_i, H_0(H_{ch_i})\big)$		
• If it does hold:	$\xrightarrow{accept}$	• Data encryption: Choose $v_i \in \mathbb{Z}_{n^2}^*$ Calculate $c_i = g^{m_i} \cdot v_i^n \mod n^2$ • Online signature generation: Choose $s_i{}' \in \mathbb{Z}_{p_1}^*$
• Maintain P_i	$\xleftarrow{P_i = ID_i \| c_i \| Ts_i \| \sigma_i^{on}}$	Calculate $\sigma_i^{on} = (s_i{}'.u_i{}')$
• Else:	$\xrightarrow{reject}$	• Revoke the aggregation command

$Y_i = g_1^{X_i}$, where $(Sig_{sk}, Ver_{pk}) = (X_i, Y_i)$. ET_i also chooses a larger random integer $k_i \in \mathbb{Z}_{p_1}^*$ as the binding factor and computes $r_i = H_1(ID_i||TS_i||k_i)$, where ID_i is the identifier of the ET_i and TS_i is the current time stamp, which can resist the potential replay attack. At last, ET_i computes the knowledge of registration $\{\alpha_i, \beta_i\}$, where $\alpha_i = g_1^{r_i}$, $\beta_i = r_i - X_i H_2(\alpha_i)$ and sends $\{Y_i, \alpha_i, \beta_i\}$ to CC.

- *Authentication:* After receiving the registration message $\{Y_i, \alpha_i, \beta_i\}$ from ET_i, CC verifies α_i by checking $\alpha_i = g_1^{\beta_i} Y_i^{H_2(\alpha_i)}$ based on discrete logarithm problem. Then, it publishes $\{Y_i, \alpha_i, \beta_i\}$.
- *Offline Signature Generation:* In order to generate the offline signature, ET_i first chooses two random values $y, z \in \mathbb{Z}_{p_1}^*$ and sets $g_2 = g_1^y$, $g_3 = g_1^z$. Without loss of generality, our LPDA-EC scheme would select the BLS short signature [22] σ_{BLS} as the secure signature scheme to generate the offline signature. ET_i also chooses two integers $(s_i, u_i) \in \mathbb{Z}_{p_1}^*$ and stores $St = (r_i, s_i, u_i)$ as the state information, where $r_i = H_1(ID_i||TS_i||k_i)$. Then, the value of DTCH function can be calculated as

$$H_{ch_i} = g_1^{r_i} \cdot g_2^{s_i} \cdot g_3^{u_i}, \tag{5}$$

and ET_i further makes a signature on H_{ch_i} as

$$\sigma_i^{BLS} = \big(H_0(H_{ch_i})\big)^{X_i} \tag{6}$$

by using the signature private key X_i. At last, ET_i sends the offline tag $T_i^{off} = (ID_i||TS_i||\sigma_i^{off})$ to the ES, where $\sigma_i^{off} = (\sigma_i^{BLS}, H_{ch_i})$ and publishes the online verification key $Ver_{on} = (g_1, g_2, g_3)$ to the CC.

C. Report Generation

Upon receiving the offline tag $T_i^{off} = (ID_i||TS_i||\sigma_i^{off})$ from ET_i, the ES first checks the time stamp TS_i and the offline signature σ_i^{off} to verify its validity. Meanwhile, ET_i needs to generate its sensing data at every certain time slot t, e.g., $t = 10$ minutes, and sends the data report to the ES. The whole offline signature verification and report generation phase includes the following steps and the detailed description is shown in Table II.

- *Offline Signature Verification:* On input the verification public key Ver_{pk} and the offline signature $\sigma_i^{off} = (\sigma_i^{BLS}, H_{ch_i})$, the offline verification algorithm is to verify whether

$$e(g_1, \sigma_i^{BLS}) = e\big(Y_i, H_0(H_{ch_i})\big). \tag{7}$$

If it does hold, the algorithm outputs *accept*; otherwise, it outputs *reject*. In order to make the offline verification efficiently, the ES can perform the batch offline verification and the correctness of verification will be presented later.

- *Data Encryption:* In our edge computing system, the ET_i will report its sensing data at every certain time slot t, e.g., $t = 10$ minutes. After the offline verification has been successfully accepted, ET_i collects the sensitive data m_i and executes the Paillier cryptographic algorithm to generate the report as

$$c_i = g^{m_i} \cdot v_i^n \mod n^2, \tag{8}$$

where v_i is a random integer in $\mathbb{Z}_{n^2}^*$.

- *Online Signature Generation:* Upon the data encryption phase has finished, ET_i chooses a random number $s_i{}' \in \mathbb{Z}_{p_1}^*$ and uses the state information $St = (r_i, s_i, u_i)$ to compute the online signature as

$$u_i{}' = \big((r_i - c_i) + (s_i - s_i{}')y + u_i z\big)z^{-1}, \tag{9}$$

where $\sigma_i^{on} = (s_i{}', u_i{}')$. At last, ET_i sends its data report $P_i = ID_i||c_i||TS_t||\sigma_i^{on}$ to the ES_j, where TS_t is the current aggregation time stamp, which can resist the replay attack.

D. Report Aggregation

After ES_j receives the total ω individual reports $\{P_1, \cdots, P_\omega\}$ from the ET in a certain time slot t, ES_j needs to check the time stamp TS_t and the online signature σ_i^{on} to verify its validity, and generate the aggregation result. The detailed description of report aggregation phase is shown in Table III.

- *Online Signature Verification:* On input the online signature σ_i^{on} and the online verification key Ver_{on}, the online verification algorithm is to verify whether

$$H_{ch}(r_i, s_i, u_i) = H_{ch}(c_i, s_i{}', u_i{}'). \tag{10}$$

TABLE III
THE DETAILED DESCRIPTION OF REPORT AGGREGATION PHASE

Edge Server		**Control Center**
• Check the time stamp: TS_t • Verify the online signnature σ_i^{on} with: $H_{ch}(r_i, s_i, u_i) = H_{ch}(c_i, {s_i}', {u_i}')$ • If it does hold: *Report aggregation:* $c = \prod_{i=1}^{\omega} c_i \mod n^2$ *Aggregation signature generation:* Choose $X_j \in \mathbb{Z}_{p_1}^*$ Calculate $\sigma_{Agg} = \big(H_0(ID_j\|c\|TS_t)\big)^{X_j}$	$\xrightarrow{P_i = ID_j\|c\|TS_t\|\sigma_{Agg}}$	• Maintain P
• Else:	$\xrightarrow{reject}$	• Revoke the aggregation command

If it does hold, the algorithm outputs *accept*; otherwise, it outputs *reject*.

- *Report Aggregation:* After the validity checking, the ES computes the aggregation results for encrypted report data as

$$c = \prod_{i=1}^{\omega} c_i \mod n^2. \tag{11}$$

- *Aggregation Signature Generation:* Then, the ES_j chooses a random number $X_j \in \mathbb{Z}_{p_1}^*$ as the aggregation signature private key, and makes an aggregation signature as

$$\sigma_{Agg} = \big(H_0(ID_j\|c\|TS_t)\big)^{X_j}, \tag{12}$$

where ID_j is the identifier of the ES_j. At last, ES_j sends the aggregated report $P = ID_j\|c\|TS_t\|\sigma_{Agg}$ to the CC.

E. Report Reading

Upon receiving $P = ID_j\|c\|TS_t\|\sigma_{Agg}$, CC performs the following steps to read the aggregated result and finally sends the response information to each ET.

- *Aggregation Signature Verification:* CC first verifies the validity of the aggregation signature σ_{Agg}, i.e., whether $e(g_1, \sigma_{Agg}) = e\big(Y_j, H_0(ID_j\|c\|TS_t)\big)$, where $Y_j = g_1^{X_j}$. If it does hold, the verification algorithm outputs *accept*, since $e(g_1, \sigma_{Agg}) = e\big(g_1, (H_0(ID_j\|c\|TS_t))^{X_j}\big) = e\big(g_1^{X_j}, H_0(ID_j\|c\|TS_t)\big) = e\big(Y_j, H_0(ID_j\|c\|TS_t)\big)$. Otherwise, it outputs *reject*.
- *Report Reading and Decryption* After the aggregation signature verification, CC reads the aggregated ciphertext c as

$$c = \prod_{i=1}^{\omega} c_i \mod n^2 = \prod_{i=1}^{\omega} g^{m_i} \cdot v_i^n \mod n^2$$
$$= g^{\sum_{i=1}^{\omega} m_i} \cdot \prod_{i=1}^{\omega} v_i^n \mod n^2 = g^m \cdot \prod_{i=1}^{\omega} v_i^n \mod n^2$$

and then obtains the aggregated plaintext as

$$m = \sum_{i=1}^{\omega} m_i = \frac{L(c^\lambda \mod n^2)}{L(g^\lambda \mod n^2)} \mod n. \tag{13}$$

F. Correctness

The correctness of user authentication, offline signature verification, and online signature verification are presented as follows.

- *User Authentication:*

$$g_1^{\beta_i} Y_i^{H_2(\alpha_i)} = g_1^{(r_i - X_i H_2(\alpha_i))} \cdot g_1^{X_i H_2(\alpha_i)}$$
$$= g_1^{r_i} = \alpha_i$$

- *Offline Batch Verification:*

$$\prod_{i=1}^{\omega} e\big(Y_i, H_0(H_{ch_i})\big) = \prod_{i=1}^{\omega} e\big(g_1^{X_i}, H_0(H_{ch_i})\big)$$
$$= \prod_{i=1}^{\omega} e\big(g_1, (H_0(H_{ch_i}))^{X_i}\big) = \prod_{i=1}^{\omega} e(g_1, \sigma_i^{BLS})$$
$$= e(g_1, \prod_{i=1}^{\omega} \sigma_i^{BLS})$$

- *Online Signature Verification:*

$$H_{ch}(c_i, {s_i}', {u_i}') = g_1^{{c_i}'} \cdot g_2^{{s_i}'} \cdot g_3^{{u_i}'}$$
$$= g_1^{c_i} \cdot (g_1^y)^{{s_i}'} \cdot g_1^{z\left((r_i - c_i) + (s_i - {s_i}')y + u_i z\right) z^{-1}}$$
$$= g_1^{c_i} \cdot (g_1^y)^{{s_i}'} \cdot g_1^{r_i} \cdot g_1^{-c_i} \cdot g_1^{y \cdot s_i} \cdot (g_1^y)^{-{s_i}'} \cdot g_1^{z \cdot u_i}$$
$$= (g_1)^{r_i} \cdot (g_1^y)^{s_i} \cdot (g_1^z)^{u_i} = g_1^{r_i} \cdot g_2^{s_i} \cdot g_3^{u_i}$$
$$= H_{ch}(r_i, s_i, u_i)$$

V. SECURITY ANALYSIS

In this section, we will discuss the security properties of our LPDA-EC scheme. In particular, following the security requirements and design goals described in section III, our analysis will focus on the authentication, confidentiality, privacy-preserving, integrity, and unforgeability.

A. Authentication

In our LPDA-EC scheme, the extended Schnorr's signature method is utilized to realize the secure authentication in registration phase. Since the Schnorr's signature method is provably secure under the discrete logarithm assumption, the ET's identity can be efficiently authenticated, where CC was assumes to be fully trusted. Specifically, we claim that an attacker cannot find a collision ${r_i}'$ to forge the knowledge of registration $\{\alpha_i, \beta_i\}$ without obtaining ID_i of ET_i, while the ET_i's real identifier ID_i is hidden in r_i by using a secure one-way hash function H_1. Even if the attacker successfully finds out the ET_i's real identifier ID_i in the process of offline

signature transmission, he also cannot obtain the hash function value r_i because the blinding factor k_i has been selected and kept secretly. Without the value of r_i, the success probability of an attacker getting the signature private key X_i in polynomial time is negligible, unless the discrete logarithm problem can be solved. Therefore, the secure authentication between ET and CC can be guaranteed. Meanwhile, the forgery attack can be resisted efficiently, which means the signatures and reports forged by attackers can be easily detected by CC in our LPDA-EC scheme.

B. Confidentiality and Privacy-Preserving

In the report generation phase, each user's private data m_i sensed by ET are encrypted as the individual ciphertext $c_i = g^{m_i} \cdot {v_i}^n \mod n^2$ by using Paillier homomorphic cryptosystem. Meanwhile, the aggregation operation uses the additive homomorphic property to aggregate the individual ciphertext c_i, which can be generically formed as

$$c = g^{(\sum_{i=1}^{\omega} m_i)} \cdot (\prod_{i=1}^{\omega} v_i)^n \mod n^2.$$

Let $m = \sum_{i=1}^{\omega} m_i$ and $v = \prod_{i=1}^{\omega} v_i$, then the aggregated ciphertext $c = g^m \cdot v^n \mod n^2$ is still a valid ciphertext of Paillier cryptosystem. Since the Paillier cryptosystem is semantically secure against the Chosen Plaintext Attack (CPA) [17], the confidentiality of both individual private data m_i and aggregated data m can be guaranteed.

Specifically, even if an attacker can monitor the whole communication channel from ET to CC, which means both the individual ciphertexts c_i and the aggregated result c can be eavesdropped by the attacker, he still cannot identify any related private information. On the one hand, after collecting all the reports from ET, the ES cannot decrypt the individual ciphertext without the private key (λ, μ) of Paillier cryptosystem, instead, it is only required to aggregate the reports directly by computing $c = \prod_{i=1}^{\omega} c_i \mod n^2$ and transmitting the aggregated results to the CC. Thus, although ES is the *honest-but-curious* entity and the attacker may intrude the database of the ES, the ET's privacy can be protected perfectly. On the other hand, upon receiving c from ES, the CC recovers it as the sum of each ET's private data $m = \sum_{i=1}^{\omega} m_i$ and stores the compressed plaintext result in the database. Even if the attacker steals this compressed plaintext result, he still cannot obtain the individual data m_i. From the analytics above, the confidentiality and privacy of each individual ET's report can be protected.

C. Integrity and Unforgeability

The proposed LPDA-EC scheme is existentially unforgeable under the chosen message attack (EU-CMA) and ensures the data integrity. According to *Definition 2*, there exists no probabilistic polynomial time adversary $\mathcal{A}$ can generate any pair (m^*, σ^*) for some $m^* \in \mathbb{Z}_{p_1}^*$ that ensures σ^* is just a valid signature on m^* with private key sk without making any query for the online signature token on m^* from the online signing oracle.

Theorem 2. *Suppose our online/offline signature scheme is (t, q_1, q_2, ϵ) secure against EU-CMA provided that we can construct an algorithm $\mathcal{B}$, which solves the q-SDH problem in polynomial time with a non-negligible probability $\epsilon' \geq \frac{\epsilon}{3} - \frac{q_2}{p}$.*

Proof. We prove this theorem by the contradiction method, assumed that $\mathcal{A}$ makes q_1 offline signature queries and makes q_2 online signature queries on message m_i. The types of successful attacks from $\mathcal{A}$ can be divided into the following cases:

Case 1: $g^{m^*} g_2^{s^*} g_3^{u^*} \neq g^{m_i} g_2^{s_i} g_3^{u_i}$ for all $i \in \{1, \cdots, q_2\}$.

Case 2: $g^{m^*} g_2^{s^*} g_3^{u^*} = g^{m_i} g_2^{s_i} g_3^{u_i}$ for some $i \in \{1, \cdots, q_2\}$, and $s^* \neq s_i$.

Case 3: $g^{m^*} g_2^{s^*} g_3^{u^*} = g^{m_i} g_2^{s_i} g_3^{u_i}$ for some $i \in \{1, \cdots, q_2\}$, and $s^* = s_i$, but $u^* \neq u_i$.

Let $\mathbb{G}$ be a cyclic group of prime order p, g be a generator of $\mathbb{G}$, and algorithm $\mathcal{B}$ is given a q-SDH instance $(g, g^{\tau}, g^{(\tau^2)}, \cdots, g^{(\tau^q)})$, its goal is to compute a new valid online/offline signature $(\sigma_{off}^*, \sigma_{on}^*)$ and successfully solve the q-SDH problem. $\mathcal{B}$ simulates a challenger $\mathcal{C}$ and interaction with adversary $\mathcal{A}$ as follows.

[CASE 1.]

- *Initiation:* $\mathcal{B}$ chooses two values $y, z \in \mathbb{Z}_p^*$ and sets signature private key as $SK = (a, y, z)$, then sends the verification public key $VK = (g, g_1, g_2, g_3)$, where $g_1 = g^a$, $g_2 = g^y$, $g_3 = g^z$ to $\mathcal{A}$.
- Sig^{off} *Queries:* $\mathcal{A}$ makes a i-th offline query, where $1 \leq i \leq q_1$. $\mathcal{B}$ randomly chooses three integers $(r_i, s_i, u_i) \in \mathbb{Z}_p^*$ to compute the value of Chameleon hash function $H_{ch_i} = g^{r_i} g_2^{s_i} g_3^{u_i} = g^{(r_i + s_i y + u_i z)}$, let $c_i = r_i + s_i y + u_i z$ and then responds with $\sigma_i^{off} = \big(H_0(H_{ch_i})^a, H_{ch_i}\big)$ as the i-th offline signature token. σ_i^{off} is sent to $\mathcal{A}$ while (r_i, s_i, u_i) are stored by $\mathcal{B}$. Obviously, σ_i^{off} is a valid offline signature for VK since
$$e\big(g, H_0(H_{ch_i})^a\big) = e\big(g_1, H_0(H_{ch_i})\big).$$
- Sig^{on} *Queries:* $\mathcal{A}$ makes a i-th online query, where $1 \leq i \leq q_2$. $\mathcal{B}$ randomly chooses ${s_i}' \in \mathbb{Z}_p^*$, sets ${u_i}' = \big((r_i - m_i) + (s_i - {s_i}')y + u_i z\big) z^{-1}$ and returns $\sigma_i^{on} = ({s_i}', {u_i}')$ as the i-th online signature token. Obviously, σ_i^{on} is a valid online signature on message m_i since
$$H_{ch_i}(r_i, s_i, u_i) = H_{ch_i}(m_i, {s_i}', {u_i}').$$
- *Forgery:* $\mathcal{A}$ finally returns a valid forgery signature $(m^*, s^*, u^*, {s_*}', {u_*}')$ satisfying the condition in *Case 1*. Since $g^{m^*} g_2^{s^*} g_3^{u^*} \neq g^{m_i} g_2^{s_i} g_3^{u_i}$, then we have $c^* = m^* + s^* y + u^* z \neq c_i$. That means $\mathcal{B}$ can generate a pair $(m^*, H_{ch}^*, \sigma^*)$ to solve the q-SDH problem in polynomial time with probability of at least $\epsilon/3$, since *Case 1* occurs with the same probability.

Note that, the simulated online/offline signing oracles of *Case 2* and *Case 3* after *Initiation*, Sig^{off} *Queries*, and Sig^{on} *Queries* phases are indistinguishable to *Case 1*. *The only difference is that algorithm $\mathcal{B}$ can compute a valid online/offline signature to solve the q-SDH problem by forging a new Chameleon hash function value H_{ch}^* in* Case 1 *whereas the trapdoor y and z are forged in* Case 2 *and* Case 3. Thus,

we skip the repeat steps to *Case 1* and focus on the *Forgery* phase in the subsequent security analysis.

[CASE 2.]

- *Forgery:* Note that in *Case 2*, the algorithm $\mathcal{B}$ will forge one of the double trapdoor y by setting the signature private key as $SK = (x, a, z)$. From the analysis above, we know that *Case 2* occurs with a probability of at least $\epsilon/3$, and $s^* = s_i$ occurs with a probability of $1/p$ since the randomly selected s_i is uniformly distributed in $\mathbb{Z}_p^*$. Thus, for the whole game the probability of $s^* = s_i$ occurring is at most q_2/p. If $\mathcal{A}$ returns a valid forgery signature $\big(m^*, \sigma_{off}^*(a, r^*, s^*, u^*), \sigma_{on}^*(s_*{}', u_*{}')\big)$ satisfying the condition in *Case 2*, which for some i, $g^{m^*} g_2^{s^*} g_3^{u^*} = g^{m_i} g_2^{s_i} g_3^{u_i}$ and $s^* \neq s_i$ hold. Then algorithm $\mathcal{B}$ can compute $a = y = \big((m^* - m_i) + (u^* - u_i)z\big)(s_i - s^*)^{-1}$. Therefore, $\mathcal{B}$ can generate a new pair (m^*, σ^*) to solve the q-SDH problem in polynomial time with probability at least $\epsilon/3 - q_2/p$.

[CASE 3.]

- *Forgery:* In *Case 3*, the algorithm $\mathcal{B}$ will forge another trapdoor z by setting $SK = (x, y, a)$, and the probability of $u^* = u_i$ occurring is at most q_2/p for the whole game. The proof is similar to that of *Case 2*, whereby $\mathcal{B}$ can compute $a = z = \big((m^* - m_i) + (s^* - s_i)z\big)(u_i - u^*)^{-1}$ for some i with probability at least $\epsilon/3 - q_2/p$ to solve the q-SDH problem in polynomial time. Here $\big(m^*, \sigma_{off}^*(a, r^*, s^*, u^*), \sigma_{on}^*(s_*{}', u_*{}')\big)$ is a valid forgery signature by $\mathcal{A}$ satisfying the condition of *Case 3*.

To sum up, we can construct an algorithm $\mathcal{B}$ which can solve the q-SDH problem in polynomial time with probability of at least $\epsilon/3 - q_2/p$. This contradicts the original q-SDH assumption and thus Theorem 2 is proved.

VI. Numerical Results

In this section, we evaluate the efficiency of our proposed scheme in terms of the computational complexity and communication overhead. We compare our LPDA-EC scheme with three existing schemes, namely, EPPA [10], PEDA [12], and SEDA [13], which are all designed from homomorphic encryption scheme. In particular, we perform several simulations to demonstrate the efficiency of our scheme. The implementation is conducted on a Linux machine with Intel Core i7-4710U CPU at 2.5GHz and 4.00 GB memory. The time cost operations are all estimated using the Pairing-Based cryptography (PBC) library. For better comparability, we choose the RSA modulus n is 1024 bits and the parameter p_1 is 160 bits. Table IV lists the notations and its time cost in our evaluations.

A. Computational Complexity

When an edge terminal ET_i joins the edge computing system, it requires two exponentiation operations in $\mathbb{Z}_{n^2}$ to generate c_i and three multiplication operations in $\mathbb{G}$ for online signature generation. After receiving the ciphertexts, the ES needs three exponentiation operations in $\mathbb{G}$ to verify the online signature and ω multiplication operations in $\mathbb{Z}_{n^2}$ to aggregate the reports. Since the multiplication operations in $\mathbb{Z}_{n^2}$ are considered negligible compared to exponentiation and pairing operations, the computational cost of aggregation is negligible. In addition, the ES also needs one exponentiation operation in $\mathbb{G}$ to generate the aggregation signature. At last, the OC performs two pairing operations and two exponentiation operations in $\mathbb{Z}_{n^2}$ to verify the validity of the aggregation signature and decrypt the aggregated ciphertext.

TABLE IV
Notations in Evaluations

Notations	Descriptions	Time Cost (ms)
T_{E_1}	Exponentiation Operation in $\mathbb{Z}_{n^2}$	1.58
T_{E_2}	Exponentiation Operation in $\mathbb{G}$	1.62
T_M	Multiplication Operation in $\mathbb{G}$	0.06
T_P	Pairing Operation	17.62

From the analysis above, we can see that there are less time-consuming cryptographic operations in our LPDA-EC scheme, especially on the ETs' side. Fig. 3(a) shows the comparison result of signature and verification time cost with other three schemes, and the detailed description of each operation is shown in Table V. From the figure, we can see that the time cost of signature and verification in our scheme is reduced at least 50% compared with EPPA [10], PEDA [12] and SEDA [13], since the time cost in their schemes rises significantly as the number of users increases. Fig. 3(b) shows the comparison results of overall computational cost among four schemes. It demonstrates that the proposed LPDA-EC is the most efficient, since the most complex operations are computed as a background computation.

TABLE V
Signature and Verification Computation Cost Comparisons

Scheme	Cost
LPDA-EC	$2T_P + (3\omega + 1)T_{E_2} + \omega T_M$
EPPA [10]	$(\omega + 3)T_P + (\omega + 1)T_M$
PEDA [12]	$(\omega + 1)T_P + (2\omega + 1)T_{E_2} + (\omega + 1)T_M$
SEDA [13]	$2T_P + (6\omega + 3)T_{E_2} + \omega T_M$

B. Communication Overhead

The communication overhead of the proposed LPDA-EC scheme includes ET-to-ES communication and ES-to-CC communication. In the ET-to-ES communication part, each ET generates the individual data report and sends it to the ES, which is in the form of $P_i = ID_i||c_i||TS_i||\sigma_i^{on}$, and its size should be $S_{ET_i} = |ID_i| + 2048 + |TS_i| + 160$, if n is 1024 bits and p_1 is 160 bits. Thus, the ES collects the total reports from ω users that are $S_{TS} = \omega S_{ET_i}$ in overall size in each certain time slot. Next, we consider the ES-to-CC communication part. In the report aggregation phase, the CC aggregates the ω individual reports and generates $P = ID_j||c||TS_t||\sigma_{Agg}$. The aggregated report form indicates that the aggregation scheme can significantly reduce the communication overhead between ES and CC. Specifically, the overhead of ES-to-CC communication decreases from $(|ID_j| + 2048 + |TS_t| + 160) * \omega$ bits to $S_{SC} = |ID_j| + 2048 + |TS_t| + 160$ bits, which means there is no correlation between ES-to-CC communication

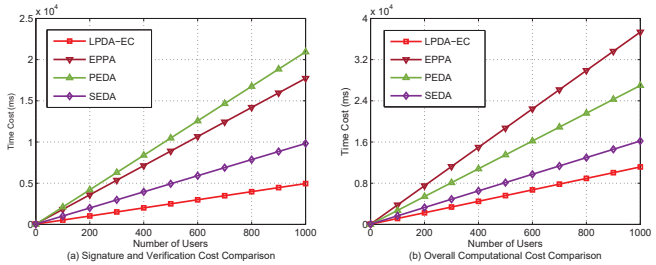

Fig. 3. Computational complexity comparison

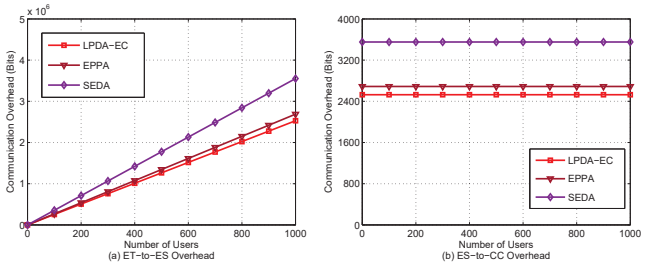

Fig. 4. Communication overhead comparison

overhead and user number. Furthermore, in Fig. 4 we plot the communication overhead in terms of the user's number ω with comparison of three schemes, where we set the size of $|ID|$ and $|TS|$ be 160 bits. Since the PEDA scheme [12] does not consider the communication overhead, we only focus on the EPPA [10], SEDA [13], and LPDA-EC. It is shown that our proposed LPDA-EC scheme is the most efficient in both ET-to-ES and ES-to-CC communication overheads by comparison with other two schemes.

VII. Conclusion

In this paper, we have proposed a lightweight privacy-preserving data aggregation scheme called LPDA-EC for edge computing system based on the online/offline signature technique, Paillier homomorphic cryptosystem and double trapdoor Chameleon hash function that can simultaneously achieve the privacy-preserving and lightweight aggregation. With the ES deployed at the network edge, LPDA-EC can transmit the time-consuming operations to the ES and minimum online computational cost. Detailed security analysis demonstrates that the proposed LPDA-EC scheme is secure under our defined security model. In addition, the extensive performance evaluations indicate the lightweight in computational costs and communication overheads. For our future work, we will extend our scheme to some specific application scenarios and consider the stronger adversarial model.

Acknowledgment

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802303, in part by the National Natural Science Foundation of China under Grant 61672283 and Grant 61602238, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20160805, and in part by the National Science Foundation grants CNS 1757533, CNS 1629746, CNS 1564128, CNS 1449860, CNS 1461932, CNS 1460971, and IIP 1439672.

References

[1] D. He, N. Kumar, S. Zeadally, A. Vinel, and L. T. Yang, "Efficient and privacy-preserving data aggregation scheme for smart grid against internal adversaries," *IEEE Transactions on Smart Grid.*, vol. 8, no. 5, pp. 2411–2419, Sep. 2017.

[2] A. M. Rahmani, T. N. Gia, B. Negash, A. Anzanpour, I. Azimi, M. Jiang, and P. Liljeberg, "Exploiting smart e-health gateways at the edge of healthcare internet-of-things: a fog computing approach," *Future Generation Computer Systems*, vol. 78, part 2, pp. 641–658, Jan. 2018.

[3] W. Ejaz, M. Naeem, A. Shahid, A. Anpalagan, and M. Jo, "Efficient energy management for the internet of things in smart cities," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 84–91, Jan. 2017.

[4] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 36–44, Jun. 2017.

[5] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[7] X. Sun and N. Ansari, "Edgeiot: Mobile edge computing for the internet of thingss," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22-29, Dec. 2016.

[8] J. Zhang, B. Chen, Y. Zhao, X. Chen, and F. Hu, "Data security and privacy-preserving in edge computing paradigm: Survey and open issues," *IEEE Access*, vol. 6, pp. 18209-18237, Mar. 2018.

[9] F. Li, B. Luo, and P. Liu, "Secure information aggregation for smart grids using homomorphic encryption," in *in Proc of IEEE SmartGridComm'10*, Gaithersburg, MD, Oct. pp. 13–16, 2012.

[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 Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, Sep. 2012.

[11] H. Li, X. Lin, H. Yang, X. Liang, R. Lu, and X. Shen, "Eppdr: an efficient privacy-preserving demand response scheme with adaptive key evolution in smart grid," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2053–2064, Aug. 2014.

[12] C.-I. Fan, S.-Y. Huang, and Y.-L. Lai, "Privacy-enhanced data aggregation scheme against internal attackers in smart grid," *IEEE Transactions on Industrial informatics*, vol. 10, no. 1, pp. 666–675, Feb. 2014.

[13] J. Ni, K. Alharbi, X. Lin, and X. Shen, "Security-enhanced data aggregation against malicious gateways in smart grid," in *Proc of IEEE GLOBECOM'15*, San Diego, CA, USA, Dec. pp. 1–6, 2015.

[14] J. Ni, K. Zhang, X. Lin, and X. Shen, "Edat: Efficient data aggregation without ttp for privacy-assured smart metering," in *Proc of IEEE ICC'16*, Kuala Lumpur, Malaysia, May. pp. 1–6, 2016.

[15] 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, Mar. 2017.

[16] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Proc of CRYPTO'01*, Santa Barbara, California, USA, Aug. pp. 213–229, 2001.

[17] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc of EUROCRYPT'99*, Czech Republic, May. pp. 223–238, 1999.

[18] C. Gao, B. Wei, D. Xie, and C. Tang, "Divisible on-line/off-line signatures," in *Proc of CT-RSA'09*, San Francisco, CA, USA, Apr. pp. 148–163, 2009.

[19] A. Shamir and Y. Tauman, "Improved online/offline signature schemes," in *Proc of CRYPTO'01*, Santa Barbara, California, USA, Aug. pp. 355–367, 2001.

[20] D. Boneh and X. Boyen, "Short signatures without random oracles," in *Proc of EUROCRYPT'04*, Interlaken, Switzerland, May. pp. 56–73, 2004.

[21] D. Catalano, M. Di Raimondo, D. Fiore, and R. Gennaro, "Off-line/on-line signatures: theoretical aspects and experimental results," in *Proc of PKC'08*, Barcelona, Spain, Mar. pp. 101–120, 2008.

[22] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *Proc of ASIACRYPT'01*, Gold Coast, Australia, Dec. pp. 514–532, 2001.