IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

Received 1 April 2013; revised 12 June 2013; accepted 29 June 2013. Date of publication 23 July 2013;
date of current version 20 September 2013.

Digital Object Identifier 10.1109/TETC.2013.2273889

PaRQ: A Privacy-Preserving Range Query Scheme Over Encrypted Metering Data for Smart Grid

MI WEN[1,2] (Member, IEEE), RONGXING LU[3] (Member, IEEE), KUAN ZHANG[2], JINGSHENG LEI[1], XIAOHUI LIANG[2] (Student Member, IEEE), AND XUEMIN SHEN[2] (Fellow, IEEE)

[1]College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai 201101, China
[2]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada
[3]School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore 639798

CORRESPONDING AUTHOR: M. WEN (wenmi2222@gmail.com)

This work was supported by the National Natural Science Foundation of China under Grants 61073189, 61272437, and 61202369, NSERC, Canada, the Innovation Program of Shanghai Municipal Education Commission under Grant 13ZZ131, the Foundation Key Project of Shanghai Science and Technology Committee under Grant 12JC1404500, and the Project of Shanghai Science and Technology Committee under Grant 12510500700.

ABSTRACT Smart grid, envisioned as an indispensable power infrastructure, is featured by real-time and two-way communications. How to securely retrieve and audit the communicated metering data for validation testing is, however, still challenging for smart grid. In this paper, we propose a novel privacy-preserving range query (PaRQ) scheme over encrypted metering data to address the privacy issues in financial auditing for smart grid. Our PaRQ allows a residential user to store metering data on a cloud server in an encrypted form. When financial auditing is needed, an authorized requester can send its range query tokens to the cloud server to retrieve the metering data. Specifically, the PaRQ constructs a hidden vector encryption based range query predicate to encrypt the searchable attributes and session keys of the encrypted data. Meanwhile, the requester's range query can be transferred into two query tokens, which are used to find the matched query results. Security analysis demonstrates that in the PaRQ, only the authorized requesters can obtain the query results, while the data confidentiality and query privacy are also preserved. The simulation results show that our PaRQ can significantly reduce communication and computation costs.

INDEX TERMS Range query, privacy, smart grid, encrypted data, metering data.

I. INTRODUCTION

Smart grid has emerged as a new concept and a promising solution for intelligent electricity generation, transmission, distribution and control [1]. The use of robust two-way communications and distributed computing technology improves the efficiency and reliability of power delivery and usage [2]. Currently, many utility companies begin to use smart grid information systems to collect real-time metering data at their control centers, via a reliable communication network deployed in parallel to the power transmission and distribution grid [3], as shown in Fig. 1. In the smart grid information system, smart meters are deployed at residential users' premises as two-way communication devices [4], [5], which periodically record the power consumption and report their metering data to a local area gateway, e.g., a wireless access point (AP). The gateway then collects and forwards data to a control center. Additionally, metering data in smart grid information systems should be periodically audited to ensure that the billing and pricing statements are presented fairly [6]. Specifically, requesters, such as market analysts, are endowed with the task of querying smart grid information systems for auditing, analysis, accounting or tax-related activities [7]. Thus, to prevent the private and sensitive information in the metering data from disclosure, data confidentiality and privacy should be achieved in financial audit for smart grid.

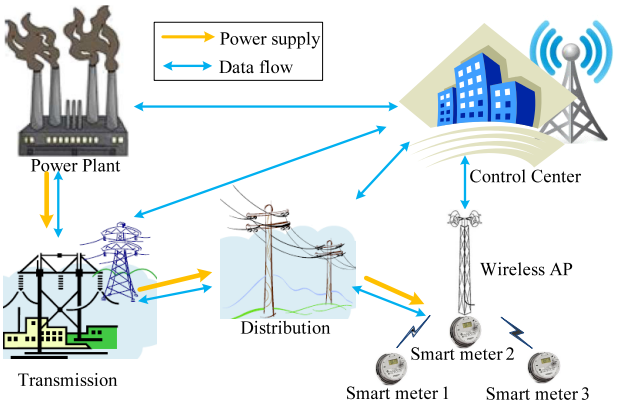

FIGURE 1. The conceptional smart grid architecture.

However, the metering data in smart grid are surging from 10,780 terabytes (TB) in 2010 to over 75,200 TB in 2015 [8], which is far beyond the control center's data management capability. Outsourcing data to cloud servers is a promising approach to relieve the control center from the burden of such a large amount of data storage and maintenance. In this approach, users can store their data on cloud servers and execute computation and queries using the servers' computational capabilities [9]. Nevertheless, cloud servers might be untrusted, and intentionally share sensitive data with the third parties for commercial purposes. Therefore, data confidentiality is important in financial audit for smart grid.

In addition, privacy concerns raise in financial auditing [10]. For instance, utility usage patterns within short intervals may reveal the users' regular daily activities [11]. In particular, data from a single house would reveal the activities of the residents, e.g., when the individual resident is at home, when he/she is watching TV [3]. If an attacker can query these data, data privacy might be violated. Therefore, users' data confidentiality and privacy should be protected and only authorized requesters can query the metering data.

From the requester's perspective, the requester, who manages the data query for financial auditing, needs to frequently query the metering data by using date ranges and/or geographic regions etc. If the query is sensitive, the requesters may prefer to keep their queries from being exposed to servers. As a result, how to operate such range queries with guaranteed query privacy is also significant for smart grid.

In this paper, we propose a Privacy-preserving Range Query (PaRQ) scheme over encrypted metering data for smart grid. The PaRQ addresses the data confidentiality and privacy problem by introducing an HVE technique. The main contributions of this paper are twofold.

- Firstly, we construct a range query predicate based on the HVE. Specifically, the session keys and the searchable attributes of the encrypted data are hidden in the HVE based range query predicate. When a requester query the cloud server, the session keys, whose encryption vectors are satisfied with the range query vectors, are released to the requester, for decrypting the encrypted metering data.
- Secondly, we analyze the security strengths and evaluate the performance of the PaRQ. Security analysis demonstrates that the PaRQ can achieve user's data confidentiality and privacy, as well as requester's query privacy. Performance evaluation results show that our PaRQ can reduce the communication and computation overhead, and shorten the response time.

The remainder of this paper is organized as follows. In Section II, we investigate the related works. In Section III, we introduce our system model, security requirements and our design goals. Then, in Section IV, we review some preliminaries. In Section V, we present our PaRQ scheme, followed by its security analysis and performance evaluation in Section VI and Section VII, respectively. Finally, we conclude this paper in Section VIII.

II. RELATED WORKS

A. SECURITY AND PRIVACY IN SMART GRID

Security and privacy are critical to the development of wireless networks [10], especially for the real-time data audit strategy in smart grid. The smart grid interpretability panel-cyber security working group [6] presents some guidelines for smart grid cyber security, including security strategy, architecture, and high-level requirements. Li [11] reviews the cyber security and privacy issues in smart grid and discusses some security and privacy solutions for smart grid. Lu et al. [3] use a super-increasing sequence to structure multidimensional data and encrypt the structured data by the holomorphic paillier cryptosystem technique. Li et al. [12] propose an authentication scheme based on merkle tree for smart grid. Acs and Castelluccia [13] exploit the privacy-preserving aggregation technique of time-series data in smart meters. They employ a differential privacy model in which users add noise to their electricity metering and the aggregator can successfully obtain the sum of the metering with a very large probability. In summary, few works focus on the query, especially range query over encrypted data in smart grid, which is really significant for user's metering data audit.

B. RANGE QUERY

Recently, the problem of querying encrypted data has been deeply investigated in both cryptography and database communities. One of the widely studied approaches is public key encryption with keyword search (PEKS) [14]. PEKS can protect users' data privacy and certain query privacy. However, most of PEKS schemes, such as the Searchable Encryption Scheme for Auction (SESA) [15], only can be applied for equality checks. Range query over the encrypted data with numeric attributes is more difficult, and most of the existing literatures cannot achieve data and query privacy simultaneously.

Roughly speaking, there are four categories of solutions that have been developed for range queries: order-preserving encryption (OPE), bucketization (Bucket), HVE and special data structure traversal. OPE-based technique [16] is to ensure that the order of plaintext data is preserved in the ciphertext domain. This allows direct translation of range predicate from the original domain to the domain of the ciphertext. However, the coupling distribution of plaintext and ciphertext domains might be exploited by attackers to guess the scope of the corresponding plaintext for a ciphertext [17]. Bucket-based technique [18] uses distributional properties of the datasets to partition and index data for efficient querying while trying to keep the information disclosure to a minimum. Queries are evaluated in an approximate manner where the returned set of records may contain some false positives.

In an HVE-based approach [19], two vectors over attributes are associated with a ciphertext and a token, respectively. Under the predicate translator, the ciphertext matches the token if and only if the two vectors are component-wise equal. Several HVE schemes [20]–[22] have been proposed in literatures. All of them use bilinear groups equipped with bilinear maps, and each constructs a proper method to hide attributes in an encrypted vector. However, it is expensive to compute exponentiation and pairing in a composite-order group. Jong [20] proposes a new HVE scheme that not only works in prime-order groups, but also requires a shorter token size and fewer pairing computations. However, Jong's scheme cannot be directly applied in the smart grid applications where data are high in dimension, variety or both.

Some specialized data structured for range query evaluation are trying to preserve notions of semantic security of the encrypted data, such as B+ tree etc. Recently, Shi et al. [23] propose a searchable encryption scheme that supports multidimensional range queries over encrypted data (MRQED). The MRQED utilizes an interval tree structure to form a hierarchical representation of intervals for each dimension and stores multiple ciphertexts corresponding to a single data value on the server, i.e., each one corresponds to a range. If it is applied to a single-dimensional data with values belonging to a domain of size N. The ciphertext representation is $O(logN)$ times the actual data. If the MRQED is applied to a piece of data with l dimensions, each query requires l times complexity to execute.

III. SYSTEM MODEL, SECURITY REQUIREMENTS AND DESIGN GOAL

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

A. SYSTEM MODEL

Our focus is on how to outsource residential users' metering data to a cloud server in encrypted form and how to operate a range query over the encrypted metering data with the help of the control center (CC). Specifically, we consider a typical residential area, as shown in Fig. 2, which is composed of a CC, two cloud servers: the CS_1 and CS_2, a requester S and some residential users $\mathbb{U} = \{U_1, U_2, \ldots, U_v\}$.

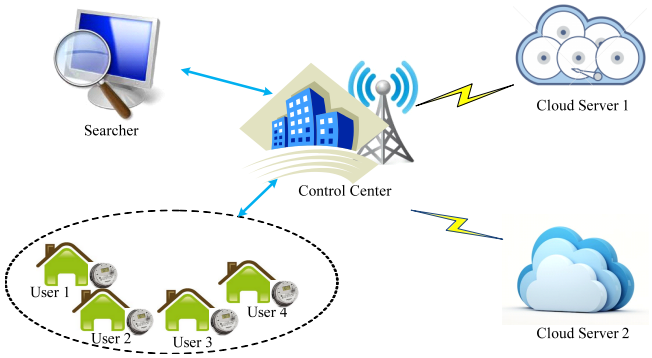

FIGURE 2. System model of PaRQ.

A residential user is the data owner, who encrypts his data by using a secret session key before outsourcing the data to the CSs. There are two cloud servers: Cloud Server 1 (CS_1) stores data ciphertexts; Cloud Server 2 (CS_2) stores session key's ciphertexts and indexes. Both servers are semi-trusted, honest but curious. We assume that either the CS_1 or CS_2 might be compromised and controlled by an adversary seeking to link users' ciphertexts with their keys, but the adversary cannot control both CSs. The control center is a trusted proxy (it operates on behalf of the utility companies), which can help users to deposit their data to cloud servers and generate query tokens for requesters to retrieve data from the servers. The requester can query the encrypted data on the cloud servers by depositing his entitling tokens to the CS_2.

The CC consists of two main components: a ciphertext forwarder, and a query translator which always operates within the secure environment. The forwarder on the CC needs to add a unique index to the data ciphertexts and the session key's ciphertexts. To preserve the query privacy, the requester's query needs to be translated into two tokens, so that the CS_2 can evaluate this query without disclosing its real value.

B. SECURITY REQUIREMENTS

We identify the security requirements for our PaRQ. In our security model, the CC is trustable, and residential users $\mathbb{U} = \{U_1, U_2, \ldots, U_v\}$ are honest as well. However, there exists an adversary $\mathcal{A}$ in the system intending to eavesdrop and invade the database on cloud servers to steal the individual users' reports. In addition, $\mathcal{A}$ can also launch some active attacks to threaten the data privacy and query privacy. Therefore, in order to prevent $\mathcal{A}$ from learning the users' data and to detect its malicious actions, the following security requirements should be satisfied in range query applications for smart grid.

- *Data Confidentiality*: The residential user can utilize symmetric or asymmetric cryptography to encrypt the data before outsourcing, and successfully prevent the unauthorized entities, including eavesdroppers

and cloud servers, from prying into the outsourced data.

- *Data privacy*: Individual residential users' data should not be accessed by unauthorized requesters. It means that only requesters with authorized query tokens can access the CS_2, and they can obtain the correct session keys when their query vectors in the tokens are satisfied with the encryption vectors. Thus, only the authorized requester can decrypt the encrypted metering data.
- *Query privacy*: As requesters usually prefer to keep their queries from being exposed to others, thus, the biggest concern is to hide their queries into tokens to protect the query privacy. Otherwise, if the query includes some sensitive information, such as "$5 \leq$ *priority* ≤ 7", then the CS_2 could know the requester is querying some important users' metering data. Then, the requester or the query results could be traced or analyzed by the curious server CS_2.

C. DESIGNING GOAL

To enable effective range query over encrypted metering data under the aforementioned model, our design goal is to develop a privacy-preserving range query scheme over encrypted data for smart grid, and to achieve the security of the data and efficient range query as follows.

- The security requirements should be guaranteed in the proposed scheme. As stated above, if the smart grid does not consider the security, the residential users' privacy could be disclosed, and the real-time power metering reports could be stealed. Therefore, the proposed scheme should achieve the data confidentiality and privacy, as well as the query privacy.
- The performance efficiency should be achieved in the proposed scheme. As range query are operated over encrypted multidimensional data, compared with existing schemes, the proposed PaRQ scheme should improve the communication, computation and response time complexities.

IV. PRELIMINARIES

In this section, we briefly describe the basic definitions and properties of bilinear pairings and HVE, which serves as the basis of the PaRQ.

A. BILINEAR PAIRING

Bilinear pairing is an important cryptographic primitive [24]. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic multiplication groups of prime order q. Let a and b be elements of Z_q^*. We assume that the discrete logarithm problem (DLP) in both $\mathbb{G}_1$ and $\mathbb{G}_2$ are hard. g is a generator of $\mathbb{G}_1$. A bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ with the following properties.

(1) Bilinear: $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G}_1^2$.
(2) Non-degenerate: $e(g, h) \neq 1_{\mathbb{G}_2}$ whenever $g, h \neq 1_{\mathbb{G}_1}$.
(3) Computable: There is an efficient algorithm to compute $e(g, h) \in \mathbb{G}_2$ for all $(g, h) \in \mathbb{G}_1^2$.

Definition 1: A bilinear parameter generator $\mathcal{G}en$ is a probabilistic algorithm that takes a security parameter κ as input, and outputs a 5-tuple $(q, g, \mathbb{G}_1, \mathbb{G}_2, e)$.

B. HEV BASED QUERY PREDICATE

The concept of HVE is proposed by Boneh and Waters [19]. HVE is a type of predicate encryption where two vectors over attributes are associated with a ciphertext and a token, respectively. At a high level, the ciphertext matches the token if and only if the two vectors are component-wise equal. There are two character sets $\sum$ and $\sum_* = \sum \cup \{*\}$ in the setting of HVE. Here $\sum$ is an arbitrary set of attributes. We assume $\sum = \mathbb{Z}_q$; $*$ is a special symbol denoting a wildcard component, which means that the component related to $*$ is not involved with any attribute. HVE mainly consists of four phases: key generation, data encryption, token generation and data query.

1) EQUALITY QUERY

- In key generation phase, the TA distributes the public/private key pair (PK, SK) to a receiver.
- In data encryption phase, a user chooses a vector $\mathbf{x} = (x_1, \ldots, x_l) \in \sum^l$ to characterize its data and encrypts its data m into a ciphertext CT using the receiver's public key.
- In token generation phase, the receiver chooses a vector $\mathbf{w} = (w_1, \ldots, w_l) \in (\sum_*)^l$ to represent his query requirements and generate a query token T_w. The receiver sends T_w to the server.
- In data query phase, if $\mathbf{x}$ equals to $\mathbf{w}$, the token can decrypt a ciphertext by using the receiver's private keys. The matching condition is defined as following: let $s(w)$ be the set of indexes i such that w_i is not a wildcard in the vector $\mathbf{w} = (w_1, \ldots, w_l)$. For the vector $\mathbf{x}$ and $\mathbf{w}$, let $P_{\mathbf{w}}(\mathbf{x})$ be the following equality predicate:

$$P_{\mathbf{w}}(\mathbf{x}) = \begin{cases} 1, & \text{if for all } i \in s(w),\ w_i = x_i, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Then, the server can disclose the data m if the equality predicate $P_{\mathbf{w}}(\mathbf{x}) = 1$.

2) COMPARISON QUERY

If we map the ith component $x_i \in \mathbf{x}$ to its domain $\{1, \ldots, n\}$ as in [20], the value of x_i is one of the number $j \in \{1, \ldots, n\}$. The key generation phase is same as in the above quality query.

Then, in the data encryption phase, the user builds an encryption vector $\sigma(\mathbf{x}) = (\sigma_{i,j}) \in \{0, 1\}^{nl}$ for $\mathbf{x} = (x_1, \ldots, x_l) \in \{1, \ldots, n\}^l$, as follows:

$$\sigma_{i,j} = \begin{cases} 1, & \text{if } x_i \geq j, \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

where, $i \in \{1, \ldots, l\}$ and $j \in \{1, \ldots, n\}$. For example, $l = 3, n = 5$ and let $\mathbf{x} = (1, 3, 2)$. Thus $\mathbf{x} = (x_1, \ldots, x_l) \in$

$\{1, \ldots, n\}^l = \{1, 2, 3, 4, 5\}^3$ and the corresponding encryption vector $\sigma(\mathbf{x}) = (10000, 11100, 11000)$. Then, the data should be encrypted under the encryption vector $\sigma(\mathbf{x})$.

Next, in the token generation phase, the user builds a query vector $\sigma^*(\mathbf{w}) = (\sigma^*_{i,j}) \in \{0, 1, *\}^{nl}$ for $\mathbf{w} = (w_1, \ldots, w_l) \in \{1, \ldots, n\}^l$ as follows:

$$\sigma^*_{i,j} = \begin{cases} 1, & \text{if } w_i = j, \\ *, & \text{otherwise.} \end{cases} \tag{3}$$

Similarly, we assume $l = 3$, $n = 5$ and $\mathbf{w} = (w_1, \ldots, w_l) \in \{1, \ldots, n\}^l = \{1, 2, 3, 4, 5\}^3$. If the receiver's query condition is $P = (x_1 \geq 1) \wedge (x_2 \geq 3) \wedge (x_3 \geq 1)$, i.e., $\mathbf{w} = (1, 3, 1)$. Thus the query vector $\sigma^*(\mathbf{w}) = (1 * * * *, * * 1 * *, 1 * * * *)$. Note that, the number of the elements in $\sigma^*(\mathbf{w})$ is nl.

In the data query phase, let $s(\sigma^*(w))$ denotes the set of all indexes k which satisfies $\sigma^*_k \neq *$, where $k \in \{1, \ldots, nl\}$. Let $P_{\sigma^*(\mathbf{w})}(\sigma(\mathbf{x}))$ be the following comparison predicate:

$$P_{\sigma^*(\mathbf{w})}(\sigma(\mathbf{x})) = \begin{cases} 1, & \text{if for all } i \in s(\sigma^*(w)), \sigma^*(w_i) = \sigma(x_i), \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

Finally, the server can disclose the data m if the comparison predicate $P_{\sigma^*(\mathbf{w})}(\sigma(\mathbf{x})) = 1$.

V. THE PROPOSED PaRQ SCHEME

In this section, we present the details of the PaRQ. There are three major phases in our scheme: construction of the range query predicate phase, encrypted data deposit phase and range query phase. Firstly, we introduce the construction of the range query predicate phase.

A. CONSTRUCTION OF THE RANGE QUERY PREDICATE

Inspired by the equality predicate and comparison predicate, we can extend them to support range query predicate. Specifically, we can achieve the opposite semantics of the above comparison query, i.e., $x_i \leq j$, by constituting the vectors $\sigma(\mathbf{x})$ in a reverse manner as Eq. (5).

$$\sigma_{i,j} = \begin{cases} 1, & \text{if } x_i \leq j, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Thus, the HVE scheme can support range queries, such as $a \leq x_i \leq b$. Table 1 illustrates notations used in this paper.

TABLE 1. The notations used in this paper.

$\sigma(\mathbf{x})$	an encryption vector which related to the ciphertexts
$\sigma^*(\mathbf{w})$	a query vector which related to the query tokens
$\sigma_{\geq}(\mathbf{x})$	an encryption vector under the predicate $x_i \geq j$
$\sigma_{\leq}(\mathbf{x})$	an encryption vector under the predicate $x_i \leq j$
$\sigma^*_{\geq}(\mathbf{w})$	a query vector of a range query where $w_i \geq a$
$\sigma^*_{\leq}(\mathbf{w})$	a query vector of a range query where $w_i \leq b$.
$s(\sigma^*_{\geq}(\mathbf{w}))$	a set of all indexes k where $\sigma^*(w_k) \neq *$
$s(\sigma^*_{\leq}(\mathbf{w}))$	a sets of all indexes k' where $\sigma^*(w_{k'}) \neq *$

The key generation phase is same as above in the quality query.

In the data encryption phase, the residential user should define two encryption vectors: $\sigma_{\geq}(\mathbf{x})$ and $\sigma_{\leq}(\mathbf{x})$ as Eq. (2) and Eq. (5) when $x_i \geq j$ and $x_i \leq j$, respectively. The receiver can obtain the correct data if and only if both conditions $x_i \geq a$ and $x_i \leq b$ hold. If the encrypted data in HVE is Ω, the residential user $U_i \in \mathbb{U}$ should split Ω into two parts by the following steps: 1) randomly chooses a polynomial $f(x) = a'x + \Omega$, where a' is a random coefficient. 2) U_i chooses two random integers and computes two data shares Ω_L and Ω_R, i.e., Ω is divided into two parts: Ω_{L} and Ω_{R}. U_i encrypts Ω_{L} and Ω_{R} under vectors $\sigma_{\geq}(\mathbf{x})$ and $\sigma_{\leq}(\mathbf{x})$, respectively.

In the token generation phase, the requester's range query are defined with two vectors: $\sigma^*_{\geq}(\mathbf{w})$ and $\sigma^*_{\leq}(\mathbf{w})$ when $w_i = a$ and $w_i = b$, respectively. Let $s(\sigma^*_{\geq}(w))$ be the set of all indexes k which satisfies $\sigma^*_{\geq}(w_k) \neq *$, and $s(\sigma^*_{\leq}(w))$ be the sets of all indexes k' which satisfy $\sigma^*_{\leq}(w_{k'}) \neq *$. Here, $k, k' \in (1, \ldots, nl)$. Finally, in the data query phase, the server checks two comparison predicates $P_{\sigma^*_{\geq}(\mathbf{w})}(\sigma_{\geq}(\mathbf{x}))$ and $P_{\sigma^*_{\leq}(\mathbf{w})}(\sigma_{\leq}(\mathbf{x}))$, which can be generated as Eq. (4). The server can obtain Ω_{L} if $\sigma_g(x_k)$ and $\sigma^*_{\geq}(w_k)$ are equal for all $k \in s(\sigma^*_{\geq}(w))$, i.e., $P_{\sigma^*_{\geq}(\mathbf{w})}(\sigma_{\geq}(\mathbf{x})) = 1$. Similarly, the server can obtain Ω_{R} if $\sigma_{\leq}(x_{k'})$ and $\sigma^*_{\leq}(w_{k'})$ are equal for all $k' \in s(\sigma^*_{\leq}(w))$, i.e., $P_{\sigma^*_{\leq}(\mathbf{w})}(\sigma_{\leq}(\mathbf{x})) = 1$. The range query predicate can be denoted as follows:

$$P_{(\sigma^*_{\geq}(\mathbf{w}), \sigma^*_{\leq}(\mathbf{w}))}(\sigma_{\geq}(\mathbf{x}), \sigma_{\leq}(\mathbf{x})) = \begin{cases} 1, & \text{if } P_{\sigma^*_{\geq}(\mathbf{w})}(\sigma_{\geq}(\mathbf{x})) = 1, \ \textit{and}, \ P_{\sigma^*_{\leq}(\mathbf{w})}(\sigma_{\leq}(\mathbf{x})) = 1 \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Finally, if $P_{(\sigma^*_{\geq}(\mathbf{w}), \sigma^*_{\leq}(\mathbf{w}))}(\sigma_{\geq}(\mathbf{x}), \sigma_{\leq}(\mathbf{x})) = 1$ the server can recover Ω_{L} and Ω_{R}. Then, the data Ω can be computed.

The main procedures of range query on encrypted data in smart grid are illustrated in Fig. 3. The CC is not only the data forwarder but also the query translator. In the encrypted data deposit phase, before outsourcing his data, a residential user U_i encrypts his data m into a ciphertext C by randomly choosing a secret session key ks. At the same time, U_i hides ks and m's searchable attributes into another ciphertext CT by using the HVE range query predicate and the CC's pubic key PK. Note that, $\Omega = ks$ in our PaRQ. Then, U_i deposits both ciphertexts C and CT to the CC. The CC adds an index Ind to both C and CT. Then the CC transmits $\{Ind, \mathsf{C}\}$ to the CS_1 and $\{Ind, \mathsf{CT}\}$ to the CS_2.

As shown in Fig. 3, when a requester S posts a range query, the query should be translated into query tokens by using the CC's private key. Then, the requester deposits its tokens to the CS_2 to retrieve the session key ks and index Ind. The session keys whose encryption vectors are satisfied with the range query vectors and their indexes can be released to the requester. The requester queries the corresponding ciphertext C from CS_1 by using its received index Ind. Then, the requester can recover the original data by using the secret key ks to decrypt C .

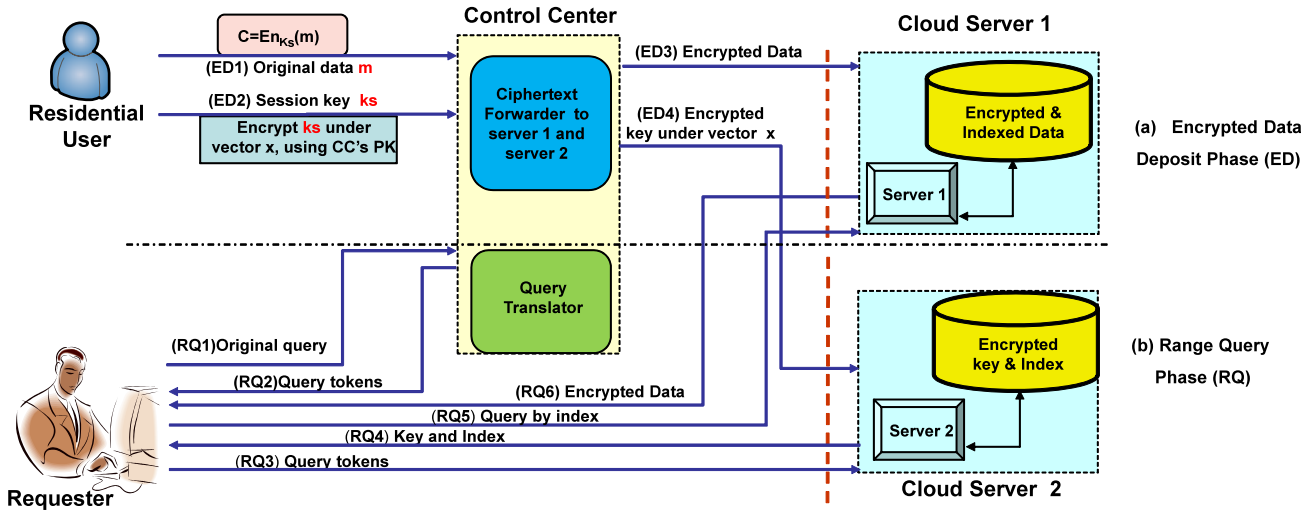

FIGURE 3. **Data query procedures.**

B. THE ENCRYPTED DATA DEPOSIT PHASE

1) KEY GENERATION

For a single-authority smart grid system, a trusted authority (TA) can bootstrap the whole system. Specifically, in the key generation phase, given the security parameters κ, TA first generates $(q, g, \mathbb{G}_1, \mathbb{G}_2, e)$ by running $\mathcal{G}en(\kappa)$. TA randomly chooses a master key $r \in Z_q^*$, and computes the corresponding public key g^r. Thus, (g^r, r) is the public/private key pair of the TA. When S applies a query, TA assigns an ID-based key pair $(H_1(ID_S), H_1^r(ID_S))$, denoted as (pk_s, sk_s), to S. TA selects some random elements g_1, g_2, $(h_1, u_1, \psi_1), \ldots, (h_{nl}, u_{nl}, \psi_{nl}) \in \mathbb{G}_1$. The TA also picks random numbers $y_1, y_2, v_1, \ldots, v_{nl}, t_1, \ldots, t_{nl} \in \mathbb{Z}_p$. Then, TA computes $Y_1 = g^{y_1}, Y_2 = g^{y_2}, v_k = g^{v_k} \in \mathbb{G}_1$ for $k \in (1, \ldots, nl)$. In addition, TA computes $\Gamma = e(g_1, Y_1)e(g_2, Y_2) \in \mathbb{G}_2$. Later, TA distributes the HVE public/private key pair (PK, SK) to the CC as follows:

$$PK = (g, Y_1, Y_2, (h_1, u_1, \psi_1, V_1, T_1), \ldots, (h_{nl}, u_{nl}, \psi_l, V_{nl}, T_{nl}))$$
$$SK = (g_1, g_2, y_1, y_2, v_1, \ldots, v_{nl}, t_1, \ldots, t_{nl}).$$

We assume that the communication channels are secure in our system model. An ID-based signature scheme $Sig(\cdot)$ [25] can be used to authenticate the requester's identity. The details of secure channel establishment is without the scope of this paper. Fig. 4 shows the dataflow of the PaRQ.

2) DATA ENCRYPTION

We denote each data as m_i. When U_i wants to report m_i to the cloud server CS_1, U_i randomly generates a session key ks_i. Then U_i encrypts its data into a ciphertext CT_i, where $\mathsf{CT}_i = \mathsf{Enc}_{ks_i}(m_i)$. $\mathsf{Enc}(\cdot)$ is a symmetric encryption algorithm, e.g., AES [26].

For each uploading interval

$$U_i \rightarrow CC : \{\mathsf{C}_i, t_1\}.$$

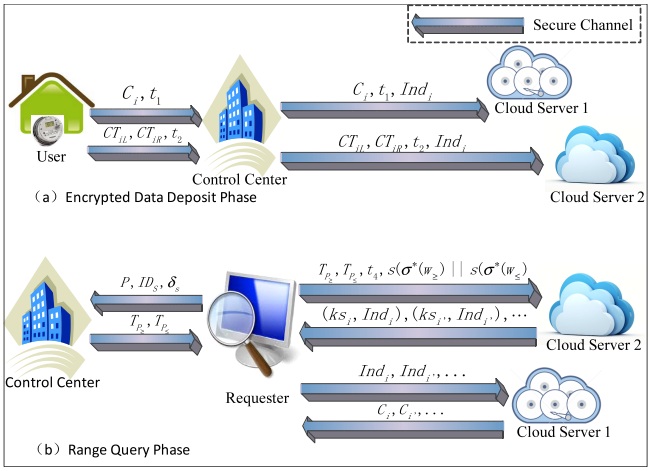

FIGURE 4. **Data flow in our PaRQ scheme.**

In this paper, "$A \rightarrow B : \{C\}$" means "A sends C to B". Then, the CC adds a unique index Ind_i to the data ciphertext and transmits all of them to the CS_1.

$$CC \rightarrow CS_1 : \{\mathsf{C}_i, t_1, Ind_i\}.$$

3) HVE-BASED SESSION KEY ENCRYPTION

If each data has l searchable attributes, U_i chooses a vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{il}) \in \sum^l$ to characterize its data m_i in different dimensions. To encrypt ks_i by using the CC's PK and the vector $\mathbf{x}_i$, U_i divides each ks_i into two parts: $ks_{i\mathsf{L}}$ and $ks_{i\mathsf{R}}$. Then, $ks_{i\mathsf{L}}$ is encrypted by using the encryption vector $\sigma_{\geq}(\mathbf{x}_i)$; $ks_{i\mathsf{R}}$ is encrypted by using the encryption vector $\sigma_{\leq}(\mathbf{x}_i)$. Thus, the CS_2 can recover ks_i only when both encryption vectors are satisfied with the corresponding query vectors in the range query tokens. The HVE-based session key encryption details are as follows:

1) Firstly, U_i maps $\mathbf{x}_i$ to an encryption vector $\sigma_{\geq}(\mathbf{x}_i)$ as Eq. (2). Then, U_i selects two random numbers $r_{i1}, r_{i2} \in$

Z_p and computes tags for the ciphertext $ks_{i\mathsf{L}}$ by using the encryption vector $\sigma_{\geq}(\mathbf{x}_i)$ as:

$$\begin{aligned}
&\mathsf{C}^i_{\mathsf{L}1} = Y_1^{r_{i1}}, \mathsf{C}^i_{\mathsf{L}2} = Y_2^{r_{i1}},\\
&\mathsf{C}^i_{\mathsf{L}3,1} = (h_1 u_1^{\sigma_{\geq}(x_{i1})})^{r_{i1}} V_1^{r_{i2}},\\
&\ldots,\\
&\mathsf{C}^i_{\mathsf{L}3,nl} = (h_{nl} u_{nl}^{\sigma_{\geq}(x_{inl})})^{r_{i1}} V_{nl}^{r_{i2}},\\
&\mathsf{C}^i_{\mathsf{L}4,1} = \psi_1^{r_{i1}} T_1^{r_{i2}},\\
&\ldots,\\
&\mathsf{C}^i_{\mathsf{L}4,nl} = \psi_{nl}^{r_{i1}} T_{nl}^{r_{i2}},\\
&\mathsf{C}^i_{\mathsf{L}5} = g^{r_{i2}}, \mathsf{C}^i_{\mathsf{L}6} = \Gamma^{r_{i1}} ks_{i\mathsf{L}}.
\end{aligned} \tag{7}$$

Let $\mathsf{CT}_{i\mathsf{L}} = (\mathsf{C}^i_{\mathsf{L}1}, \mathsf{C}^i_{\mathsf{L}2}, \mathsf{C}^i_{\mathsf{L}3,1}, \ldots, \mathsf{C}^i_{\mathsf{L}3,nl}, \mathsf{C}^i_{\mathsf{L}4,1}, \ldots, \mathsf{C}^i_{\mathsf{L}4,nl}, \mathsf{C}^i_{\mathsf{L}5}, \mathsf{C}^i_{\mathsf{L}6})$.

2) Secondly, U_i maps $\mathbf{x}_i$ to an encryption vector $\sigma_{\leq}(\mathbf{x}_i)$ as Eq. (5). Then, U_i selects two random numbers r'_{i1}, $r'_{i2} \in Z_p$, and computes tags for the ciphertext $ks_{i\mathsf{R}}$ by using the encryption vector $\sigma_{\leq}(\mathbf{x}_i)$:

$$\begin{aligned}
&\mathsf{C}^i_{\mathsf{R}1} = Y_1^{r'_{i1}}, \mathsf{C}^i_{\mathsf{R}2} = Y_2^{r'_{i1}},\\
&\mathsf{C}^i_{\mathsf{R}3,1} = (h_1 u_1^{\sigma_{\leq}(x_{i1})})^{r'_{i1}} V_1^{r'_{i2}},\\
&\ldots,\\
&\mathsf{C}^i_{\mathsf{R}3,nl} = (h_{nl} u_{nl}^{\sigma_{\leq}(x_{inl})})^{r'_{i1}} V_{nl}^{r'_{i2}},\\
&\mathsf{C}^i_{\mathsf{R}4,1} == \psi_1^{r'_{i1}} T_1^{r'_{i2}},\\
&\ldots,\\
&\mathsf{C}^i_{\mathsf{R}4,nl} = \psi_{nl}^{r'_{i1}} T_{nl}^{r'_{i2}},\\
&\mathsf{C}^i_{\mathsf{R}5} = g^{r'_{i2}}, \mathsf{C}^i_{\mathsf{R}6} = \Gamma^{r'_{i1}} ks_{i\mathsf{R}}.
\end{aligned} \tag{8}$$

Let $\mathsf{CT}_{i\mathsf{R}} = (\mathsf{C}^i_{\mathsf{R}1}, \mathsf{C}^i_{\mathsf{R}2}, \mathsf{C}^i_{\mathsf{R}3,1}, \ldots, \mathsf{C}^i_{\mathsf{R}3,nl}, \mathsf{C}^i_{\mathsf{R}4,1}, \ldots, \mathsf{C}^i_{\mathsf{R}4,nl}, \mathsf{C}^i_{\mathsf{R}5}, \mathsf{C}^i_{\mathsf{R}6})$.

4) CIPHERTEXT DEPOSIT

U_i deposits $\mathsf{CT}_{i\mathsf{L}}$ and $\mathsf{CT}_{i\mathsf{R}}$ to the CC as:

$$U_i \rightarrow CC : \{\mathsf{CT}_{i\mathsf{L}}, \mathsf{CT}_{i\mathsf{R}}, t_2\}.$$

The CC also adds the index Ind_i to the key ciphertext and transmits all of them to the CS_2.

$$CC \rightarrow CS_2 : \{\mathsf{CT}_{i\mathsf{L}}, \mathsf{CT}_{i\mathsf{R}}, t_2, Ind_i\}.$$

C. RANGE QUERY PHASE

1) TOKEN GENERATION

When S wants to query the server to retrieve the expected data, S firstly generates a range query, such as $P = (a_1 \leq x_1 \leq b_1) \wedge (a_2 \leq x_2 \leq b_2) \cdots \wedge (a_l \leq x_l \leq b_l)$. S then computes a signature $\delta_s = Sig_{sk_s}(P)$.

In each querying interval

$$S \rightarrow CC : \{P, ID_S, \delta_s\}.$$

The CC verifies S's signature by using S's public key. If S is an authorized requester, the CC might issue query tokens to S by following steps. The CC divides P into two parts: $P_{\geq} = (x_1 \geq a_1) \wedge (x_2 \geq a_2) \cdots \wedge (x_l \geq a_l)$ and $P_{\leq} = (x_1 \leq b_1) \wedge (x_1 \leq b_2) \cdots \wedge (x_1 \leq b_l)$. Let $w_{\geq} = (a_1, \ldots, a_l)$ and $w_{\leq} = (b_1, \ldots, b_l)$.

The CC generates two query vector $\sigma^*_{\geq}(\mathbf{w})$ and $\sigma^*_{\leq}(\mathbf{w})$ to represent $P_{\geq}$ and $P_{\leq}$ as Eq. (3), respectively. The wildcard $*$ in the vector $\sigma^*_{\geq}(\mathbf{w})$ means that S does not care about the attributes related to $*$. Let $s(\sigma^*(w_g))$ be the set of k which satisfies $\sigma^*_{\geq}(w_k) \neq *$. Let $s(\sigma^*_{\leq}(w))$ be the set of k' which satisfies $\sigma^*_{\leq}(w_{k'}) \neq *$. Then the CC computes a token $T_{P_{\geq}}$ by using the query vector $\sigma^*_{\geq}(\mathbf{w})$ as follows.

1) Select a random $\alpha, \beta \in Z_p$, and generate $\lambda_k, \psi_k, \gamma_k, \tau_k \in Z_p$ such that $\lambda_k y_1 + \psi_k y_2 = \alpha, \gamma_k y_1 + \tau_k y_2 = \beta$ for all $k \in s(\sigma^*_{\geq}(w))$.
2) Compute a token $T_{P_{\geq}}$ as

$$\begin{aligned}
&K^s_{\mathsf{L}1} = g_1 \prod_{k \in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma^*_{\geq}(w_k)})^{\lambda_k} \psi_k^{\gamma_k},\\
&K^s_{\mathsf{L}2} = g_2 \prod_{k \in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma^*_{\geq}(w_k)})^{\varphi_k} \psi_k^{\tau_k},\\
&K^s_{\mathsf{L}3} = g^{\alpha},\\
&K^s_{\mathsf{L}4} = g^{\beta},\\
&K^s_{\mathsf{L}5} = g^{-\sum_{k \in s(\sigma^*_{\geq}(w))} (v_k \alpha + t_k \beta)}.
\end{aligned} \tag{9}$$

Similarly, the CC generates a token $T_{P_{\leq}}$ by using the query vector $\sigma^*_{\leq}(\mathbf{w})$ as follows.

1) Select a random $\alpha', \beta' \in Z_p$, and generate $\lambda_{k'}, \psi_{k'}, \gamma_{k'}, \tau_{k'} \in Z_p$ such that $\lambda_{k'} y_1 + \psi_{k'} y_2 = \alpha'$, $\gamma_{k'} y_1 + \tau_{k'} y_2 = \beta'$ for all $k' \in s(\sigma^*_{\leq}(w))$.
2) Compute the token $T_{P_{\leq}}$ as

$$\begin{aligned}
&K^s_{\mathsf{R}1} = g_1 \prod_{k' \in s(\sigma^*(w_{ls}))} (h_{k'} u_{k'}^{\sigma^*(w_{lsk'})})^{\lambda_{k'}} \psi_{k'}^{\gamma_{k'}},\\
&K^s_{\mathsf{R}2} = g_2 \prod_{k' \in s(\sigma^*(w_{ls}))} (h_{k'} u_{k'}^{\sigma^*(w_{lsk'})})^{\varphi_{k'}} \psi_{k'}^{\tau_{k'}},\\
&K^s_{\mathsf{R}3} = g^{\alpha'},\\
&K^s_{\mathsf{R}4} = g^{\beta'},\\
&K^s_{\mathsf{R}5} = g^{-\sum_{k' \in s(\sigma^*(w_{ls}))} (v_{k'} \alpha' + t_{k'} \beta')}.
\end{aligned} \tag{10}$$

Let $T_{P_{\geq}} = (K^s_{\mathsf{L}1}, K^s_{\mathsf{L}2}, K^s_{\mathsf{L}3}, K^s_{\mathsf{L}4}, K^s_{\mathsf{L}5})$ and $T_{P_{\leq}} = (K^s_{\mathsf{R}1}, K^s_{\mathsf{R}2}, K^s_{\mathsf{R}3}, K^s_{\mathsf{R}4}, K^s_{\mathsf{R}5})$. Then, the CC keeps a record $(ID_s, T_{P_{\geq}}, T_{P_{\leq}})$ in its database and distributes $T_{P_{\geq}}$ and $T_{P_{\leq}}$ to the requester S as its authorized tokens:

$$CC \rightarrow S : \{T_{P_{\geq}}, T_{P_{\leq}}\}.$$

2) KEY AND INDEX QUERY

After receiving the query tokens from the CC, the requester deposits them as well as the non-wildcard indexes sets to the cloud server CS_2 as:

$$S \rightarrow CS_2 : \{T_{P_{\geq}}, T_{P_{\leq}}, t_4, s(\sigma^*_{\geq}(w)), s(\sigma^*_{\leq}(w))\}.$$

Then, the CS_2 searches its database to find whether there is a key ciphertext which matches the requester's query conditions. For each key ciphertext, if its encryption vectors are

satisfied with the query vectors in the query tokens, the CS_2 can obtain:

$$ks_{i\mathsf{L}} = \frac{D_1.D_2.e(K^s_{\mathsf{L}5}, \mathsf{C}^i_{\mathsf{L}5}).\mathsf{C}^i_{\mathsf{L}6}}{e(K^s_{\mathsf{L}1}, \mathsf{C}^i_{\mathsf{L}1}).e(K^s_{\mathsf{L}2}, \mathsf{C}^i_{\mathsf{L}2})}, \tag{11}$$

$$ks_{i\mathsf{R}} = \frac{D'_1.D'_2.e(K^s_{\mathsf{R}5}, \mathsf{C}^i_{\mathsf{R}5}).\mathsf{C}^i_{\mathsf{R}6}}{e(K^s_{\mathsf{R}1}, \mathsf{C}^i_{\mathsf{R}1}).e(K^s_{\mathsf{R}2}, \mathsf{C}^i_{\mathsf{R}2})}, \tag{12}$$

where $D_1 = e(K^s_{\mathsf{L}3}, \prod_{k\in s(\sigma^*_{\geq}(w))} \mathsf{C}^i_{\mathsf{L}3,k})$ and $D_2 = e(K^s_{\mathsf{L}4}, \prod_{k\in s(\sigma^*_{\geq}(w))} \mathsf{C}^i_{\mathsf{L}4,k})$. $D'_1 = e(K^s_{\mathsf{R}3}, \prod_{k'\in s(\sigma^*_{\leq}(w))} \mathsf{C}^i_{\mathsf{R}3,k'})$ and $D'_2 = e(K^s_{\mathsf{R}4}, \prod_{k'\in s(\sigma^*_{\leq}(w))} \mathsf{C}^i_{\mathsf{R}4,k'})$. Thus, the CS_2 can obtain ks_i by using $ks_{i\mathsf{L}}$ and $ks_{i\mathsf{R}}$. Then, the CS_2 sends this recovered key ks_i with indexes to the requester.

$$CS_2 \rightarrow S : \{(ks_i, Ind_i), (ks_{i'}, Ind_{i'}), \ldots\}.$$

Note that, there might be more than one session keys which are satisfied with the range query tokens. Then, the CS_2 distributes all of session keys back to the requester S.

3) DATA QUERY

Upon receiving the keys and indexes from the CS_2, the requester queries the CS_1 by using the received indexes to obtain the corresponding data ciphertext.

$$S \rightarrow CS_1 : \{Ind_i, Ind_{i'}, \ldots\}.$$

Then, the CS_1 searches its database to find whether there are ciphertexts matching the requester's indexes. If so, the CS_1 sends matched ciphertexts to the requester S:

$$CS_1 \rightarrow S : \{\mathsf{C}_i, \mathsf{C}_{i'}, \ldots\}.$$

4) DATA DECRYPTION

After receiving the real session keys from the CS_2 and ciphertexts $\{\mathsf{C}_i, \mathsf{C}_{i'}, \ldots\}$ from the CS_1, the requester S can obtain the real data by using the session keys to decrypt the ciphertexts, otherwise, C_i can be discarded.

$$S : m_i = \mathsf{Dec}_{ks_i}(\mathsf{C}_i), \ldots$$

$\mathsf{Dec}(\cdot)$ is the symmetric decryption algorithm corresponding to the opposite operation of $\mathsf{Enc}(\cdot)$.

D. ENHANCEMENT WITH COLLUSION RESILIENCE

In our system model, we assume that the adversary cannot control both CSs. Actually, in order to prevent the cloud server CS_1 and CS_2 in collusion to disclose the data m_i, an identity based encryption scheme [24] can be used in the data encryption phase to encrypt m_i. For instance, the CC has one pair of identity based public/private key (pk, sk). Firstly, m_i is encrypted by the CC's public key pk. Againit is encrypted by using the session key ks_i and be outsourced to the CS_1. Thus, even if the CS_2 recovers the session key ks_i with the requester's query tokens, the CS_2 can not decrypt the ciphertexts on the CS_1 to obtain the data m_i. The reason is that the CS_2 cannot obtain the CC's ID-based private key sk. When an authorized requester S asks for query tokens from the CC, the CC replies the query tokens as well as its identity based private key sk. Accordingly, the requester can recover data m_i by using both session key ks_i from CS_1 and sk from the CC. Note that, the requesters are high-level users, such as energy company's financial auditors, who are authorized by the CC. Therefore, they can obtain the CC's private key to decrypt their queried data. As a result, the PaRQ can remain secure even the two cloud servers are in collusion.

Furthermore, to provide forward security and prevent requesters from decrypting future encrypted data by using CC's old private key, the CC can compute different pairs of time and identity based public/private key pairs at different time intervals. Therefore, the authorized requesters can only obtain CC's private keys corresponding to their entitled intervals to decrypt their required data.

VI. SECURITY ANALYSIS

In this section, we analyze the security properties of the proposed PaRQ according to the security requirements discussed in section III.B.

- *The individual residential users' data confidentiality can be achieved.* In the PaRQ, the residential user's data m_i is encrypted by its session key ks_i. For the eavesdroppers and the CS_2, they cannot obtain anything from the ciphertext C_i because they lack of the secret key ks_i. Although the CS_2 can recover ks_i from the requester's query tokens, the CS_2 cannot extract the ciphertexts from the CS_1 if they are not in collusion. Our enhancement introduced in section V.D can be resilient to the collusion attack even if the CS_1 is in collusion with the CS_2. Accordingly, the individual residential users data confidentiality is achieved in the proposed PaRQ scheme.
- *The individual residential users' data privacy can be preserved.* In our PaRQ, ks_i and the searchable attributes are hid in two ciphertexts $\{\mathsf{CT}_{i\mathsf{L}}, \mathsf{CT}_{i\mathsf{R}}\}$. Other requesters cannot obtain ks_i from the CS_2 if they cannot obtain the authorized query tokens $\{T_{P_{\geq}}, T_{P_{\leq}}\}$ from the CC or their query vectors in the query tokens are not satisfied with the encryption vectors on the ciphertexts. Accordingly, they cannot pry into the session keys and decrypt the encrypted metering data. The range query correctness can be demonstrated as follows. In Eq. (11), the numerator equals:

$$\begin{aligned} &D_1 \cdot D_2 \cdot e(K^s_{\mathsf{L}5}, \mathsf{C}^i_{\mathsf{L}5}) \cdot \mathsf{C}^i_{\mathsf{R}6} \\ &= e\left(g^{\alpha}, \prod_{k\in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma_{\geq}(x_{ik})})^{r_{i1}} g^{v_k r_{i2}}\right) \\ &\quad \cdot e\left(g^{\beta}, \prod_{k\in s(\sigma^*_{\geq}(w))} \psi_1^{r_{i1}} g^{t_1 r_{i2}}\right) \\ &\quad \cdot e\left(g^{-\sum_{k\in s(\sigma^*_{\geq}(w))}(v_k\alpha + t_k\beta)}, g^{r_{i2}}\right) \cdot \Gamma^{r_{i1}} ks_{i\mathsf{L}} \end{aligned}$$

$$= e\left(g^{\alpha}, \prod_{k\in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma_{\geq}(x_{ik})})^{r_{i1}}\right)$$
$$\cdot e\left(g^{\beta}, \prod_{k\in s(\sigma^*_{\geq}(w))} \psi_k^{r_{i1}}\right)$$
$$\cdot e\left(g^{r_{i2}}, \prod_{k\in s(\sigma^*_{\geq}(w))} g^{v_k\alpha+t_k\beta}\right)$$
$$\cdot e\left(g^{-\sum_{k\in s(\sigma^*_{\geq}(w))}(v_k\alpha+t_k\beta)}, g^{r_{i2}}\right)\cdot \Gamma^{r_{i1}} ks_{i\mathsf{L}}$$
$$= e\left(g^{\alpha}, \prod_{k\in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma_{\geq}(x_{ik})})^{r_{i1}}\right)$$
$$\cdot e\left(g^{\beta}, \prod_{k\in s(\sigma^*_{\geq}(w))} \psi_k^{r_{i1}}\right)\cdot \Gamma^{r_{i1}} ks_{i\mathsf{L}}. \quad (13)$$

while the denominator equals:

$$e(K^s_{\mathsf{L}1}, \mathsf{C}^i_{\mathsf{L}1}).e(K^s_{\mathsf{L}2}, \mathsf{C}^i_{\mathsf{L}2})$$
$$= e\left(g_1 \prod_{k\in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma^*_{\geq}(w_k)})^{\lambda_k}\psi_k^{\gamma_k}, g^{y_1 r_{i1}}\right)$$
$$\cdot e\left(g_2 \prod_{k\in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma^*_{\geq}(w_k)})^{\varphi_k}\psi_k^{\tau_k}, g^{y_2 r_{i1}}\right)$$
$$= \Gamma^{r_{i1}}\cdot \prod_{k\in s(\sigma^*_{\geq}(w))} [e((h_k u_k^{\sigma^*_{\geq}(w_k)})^{\lambda_k}, g^{y_1 r_{i1}})$$
$$\cdot e((h_k u_k^{\sigma^*_{\geq}(w_k)})^{\varphi_k}, g^{y_2 r_{i1}})]$$
$$\cdot \prod_{k\in s(\sigma^*_{\geq}(w))} [e(\psi_k^{\gamma_k}, g^{y_1 r_{i1}})\cdot e(\psi_k^{\tau_k}, g^{y_2 r_{i1}})]$$
$$= \Gamma^{r_{i1}}\cdot \prod_{k\in s(\sigma^*_{\geq}(w))} e((h_k u_k^{\sigma^*_{\geq}(w_k)})^{r_{i1}}, g^{\lambda_k y_1+\psi_k y_2})$$
$$\cdot \prod_{k\in s(\sigma^*_{\geq}(w))} e(\psi_k^{r_{i1}}, g^{\gamma_k y_1+\tau_k y_2})$$
$$= \Gamma^{r_{i1}}\cdot e\left(\prod_{k\in s(\sigma^*_{\geq}(w))} (h_k u_k^{\sigma^*_{\geq}(w_k)})^{r_{i1}}, g^{\alpha}\right)$$
$$\cdot e\left(\prod_{k\in s(\sigma^*_{\geq}(w))} \psi_k^{r_{i1}}, g^{\beta}\right). \quad (14)$$

Let Θ be the set of indexes $i \in s(\sigma^*_{\geq}(w))$ where $\sigma^*_{\geq}(w_k) \neq \sigma_{\geq}(x_{ik})$. The Eq. (11) outputs follows

$$\frac{D_1\cdot D_2\cdot e(K_{s50}, C_{i50})\cdot C_{i60}}{e(K_{s10}, C_{i10})\cdot e(K_{s20}, C_{i20})}$$
$$= e(g^{\alpha}, \prod_{k\in\Theta}(u_k^{\sigma_{\geq}(x_{ik})-\sigma^*_{\geq}(w_k)})^{r_{i1}})\cdot ks_{iL}$$
$$= e(g, g)^{\alpha r_{i1}\Sigma_{k\in\Theta}(log_g(u_k))(\sigma_g(x_{ik})-\sigma^*_{\geq}(w_k))}. \quad (15)$$

If $\sigma^*_{\geq}(w_k)$ equals $\sigma_{\geq}(x_{ik})$ for all $k \in s(\sigma^*_{\geq}(w))$, $ks_{i\mathsf{L}}$ can be recovered; otherwise, the unauthorized requesters cannot obtain $ks_{i\mathsf{L}}$ according to Eq. (11).

Similarly, the unauthorized requesters cannot obtain $ks_{i\mathsf{R}}$ from Eq. (12). Therefore, only the authorized requester can obtain the query results and the users' data privacy is preserved.

- *The requester's query privacy can be preserved.* In our PaRQ, a requester's query is divided into two parts: $P_{\geq}$ and $P_{\leq}$, by the CC. Both of them are translated into tokens $T_{P_{\geq}}$ and $T_{P_{\leq}}$ in the form of $\prod_{k\in s(\sigma^*_{\geq}(w))}(h_k u_k^{\sigma^*_{\geq}(w_k)})^{\lambda_k}\psi_k^{\gamma_k}$ and $\prod_{k'\in s(\sigma^*_{\leq}(w))}(h_{k'} u_{k'}^{\sigma^*_{\leq}(w_{k'})})^{\varphi_{k'}}\psi_{k'}^{\tau_{k'}}$, respectively. For the eavesdroppers, they can learn nothing about the query P only with the tokens $T_{P_{\geq}}$ and $T_{P_{\leq}}$ because they have no idea about the index sets and encryption parameters $\lambda_k, \psi_k, \gamma_k, \tau_k$. Since the CS_2 still does not know the encryption parameters $\lambda_k, \psi_k, \gamma_k, \tau_k$, it also cannot obtain the real value of P even with the tokens and the index sets $s(\sigma^*_{\geq}(w))$ and $s(\sigma^*_{\leq}(w))$. Therefore, the requester's query privacy is preserved in the proposed PaRQ scheme.

From the above security analysis and comparison in Table 2, our PaRQ can achieve all of the data confidentiality and privacy and query privacy, compared with order-preserving encryption (OPE) [16] based technique and bucketization (Buket) based technique [18].

TABLE 2. Comparison of security properties.

Properties	PaRQ	MRQED[23]	Buket [18]	OPE[16]
Confidentiality	Yes	Yes	Yes	Yes
Data Privacy	Yes	Yes	partial	partial
Query Privacy	Yes	No	No	No

VII. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed PaRQ scheme in terms of the communication overhead, computation complexity and response time of the system.

A. COMMUNICATION OVERHEAD

We numerically analyze the communication overhead of our PaRQ, compared with the MRQED [23], in terms of the public key size, ciphertexts size and token size. Since the functionality of the decryption key computation phase in determining the query results in MRQED is similar to that of query tokens in the PaRQ, therefore, we take their size in comparison. "Tok/Dek" in Table 2 and Table 3 represents Token or Decryption key. Since most pairing-based cryptosystems need to work in a subgroup of the elliptic curve $E(F_q)$, by representing elliptic curve points using point compression, the lengths of the elements in G_1 and G_2 are roughly 161-bit (using point compression) and 1,024-bit, respectively. In the following, l is the number of data dimensions and N is domain of attribute values.

TABLE 3. Comparison of communication complexity (bit).

Communication	PaRQ	MRQED [23]
Public key	$(5Nl+3)\times 161$	$8Nl\times 161+1024$
Ciphertext	$(4Nl+6)\times 161+2304$	$(4Nl+1)\times 161+1024$
Tok/Dek	1610	$5Nl\times 161$

TABLE 4. Comparison of computation complexity.

Computation	PaRQ	MRQED [23]
Encryption	$(10Nl+8)C_e$	$(8Nl+3)C_e$
Tok/Dek Generation	$(12l+6)C_e$	$8NlC_e$
Query in Database	$10C_p$	$5l\cdot l_{og}\mathrm{N}C_p$

In the key generation phase, the public key includes $(5Nl+3)$ G_1 elements. If we choose AES ciphertext with 256-bit, the data ciphertext C_i is only of 256 bits. In addition, session key's ciphertext includes two parts: $\mathsf{CT}_{i\mathsf{L}}$ and $\mathsf{CT}_{i\mathsf{R}}$, each of which includes $(2Nl+3)$ G_1 elements and a G_2 element, thus the size of each session key's ciphertext is $(4Nl+6)\times 161+2048$ bits. Since each query token includes 5 G_1 elements, the size of the two query tokens $\{T_{P_{\geq}}, T_{P_{\leq}}\}$ is $161\times 10=1610$ bits.

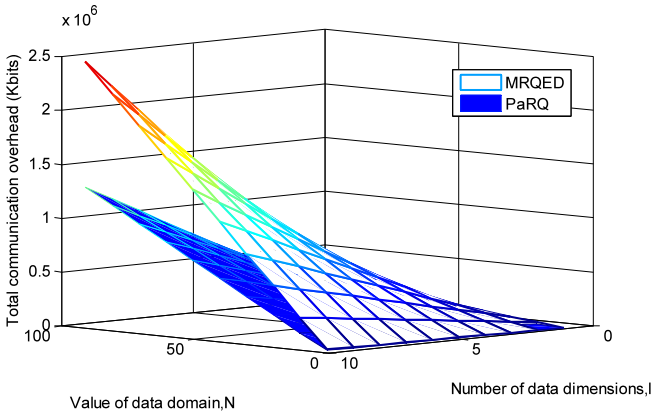

FIGURE 5. Comparison of communication overhead between PaRQ and MRQED schemes.

In comparison, the public key in the MRQED [23] includes $8Nl$ G_1 elements and a G_2 element. The decryption keys include $5Nl$ G_1 elements. In addition, there are $(4Nl+1)$ G_1 elements and a G_2 element in the ciphertexts. Table 3 shows that compared with the MRQED, our PaRQ consumes less communication overhead. Especially, our PaRQ significantly reduces the tokens transmission overhead, which is a constant, i.e., 1600 bits; in the MRQED the transmission overhead of the decryption keys may increase with both l and N. The total communication overhead comparison is depicted in Fig. 5. It further indicates that our PaRQ costs less communication overhead than the MRQED.

B. COMPUTATION OVERHEAD

In our PaRQ, the computation tasks include pairing operations and exponentiation operations. For simplicity of description, the pairing operation and exponentiation operation are denoted as C_p and C_e, respectively. Since the AES encryption/decryption and multiplication are much faster than the pairing operations, we do not analyze the AES encryption/decryption and multiplication in this subsection.

For the PaRQ, the symmetric encryption of C_i is very fast. Meanwhile, the corresponding session key is encrypted into key ciphertexts $\{\mathsf{CT}_{i0}, \mathsf{CT}_{i1}\}$ by using its encryption vectors. The computation overhead of $\{\mathsf{CT}_{i0}, \mathsf{CT}_{i1}\}$ is $(10Nl+8)C_e$ because each part requires $(5Nl+4)$ exponentiation operations. In the token generation phase, the computation cost of $\{T_{P_{\geq}}, T_{P_{\leq}}\}$ is $(12l+2)C_e$, because each query token in $\{T_{P_{\geq}}, T_{P_{\leq}}\}$ needs $6l+3$ exponentiation operations. After receiving the tokens, the CS_2 needs to compute 10 pairings to recover the session key $\{ks_{i\mathsf{L}}, ks_{i\mathsf{R}}\}$, i.e., $10C_p$.

On the other hand, the MRQED [23] needs $(8Nl+3)$ exponentiation operations to encrypt a message, another $8Nl$ exponentiation operations to derive the decryption keys and $5l\cdot l_{og}N$ pairing operations to search the correct results. From Table 4, we can see that the encryption overhead in both PaRQ and MRQED increase with l and N. The computation overhead of token generation in the PaRQ only increases wtih l, whereas, the overhead of decryption key generation in MRQED increases with both l and N. When a query is executed in a database, the overhead in our PaRQ is a constant $(10C_p)$; the overhead in the MRQED still increases with both l and N. Hence, our PaRQ is much more efficient than the MRQED. Further comparison of their range query response time is analyzed in following subsection.

C. RESPONSE TIME

To provide good services to requesters, the response time of a range query is an important metric. For example, it would be useful for the requesters to know how long they exactly need to wait for a range query result so that they can efficiently schedule their tasks. Actually, response time varies according to many factors, such as communication latency etc. We analyze the response time of our PaRQ with or without considering the network communication latency Δ. The other factors are not being included in this calculation of the response time.

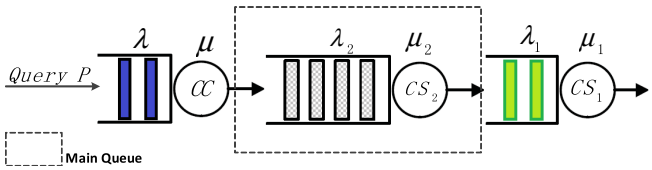

FIGURE 6. Tandem model [27] of a range query process.

In the PaRQ, a range query is processed by the CC, CS_1 and CS_2. We model our range query process as a tandem model of network queues [27], as shown in Fig. 6. We assume that the range query arrives the system according to a poisson process with rate λ, and uses the CC for token generation in an exponentially distributed time interval with mean $1/\mu$ (as an

$M/M/1$ queue). Upon exiting the CC, the requester continue accesses the CS_2 with rate λ_2 for a time which is deterministic $1/\mu_2$ (as an $M/D/1$ queue). Finally, the requester accesses server CS_1 with rate λ_1 for a time which is exponentially distributed with mean $1/\mu_1$ (as an $M/M/1$ queue). Let

$$\rho = \frac{\lambda}{\mu}; \rho_1 = \frac{\lambda_1}{\mu_1}; \rho_2 = \frac{\lambda_2}{\mu_2}.$$

If all the network states are in the set of $n = \{n_0, n_1, n_2\}$, according to Jackson's Theorem [28], the steady-state probability distribution of the system is given as:

$$P(n_1, n_2, n_3) = \rho^{n_0}(1-\rho)\rho_1^{n_1}(1-\rho_1)\rho_2^{n_2}(1-\rho_2).$$

Let T, T_1 and T_2 be the average queuing delay of the CC, CS_1 and CS_2, respectively. They can be calculated as:

$$T = \frac{1}{\mu-\lambda} = \frac{1}{\mu(1-\rho)};$$
$$T_1 = \frac{1}{\mu_1-\lambda_1} = \frac{1}{\mu_1(1-\rho_1)};$$
$$T_2 = \frac{1}{\mu_2}\frac{2-\rho_2}{2-2\rho_2}.$$

Then, the total delay of the range query in the PaRQ is:

$$T_{tol} = \frac{1}{\mu(1-\rho)} + \frac{1}{\mu_1(1-\rho_1)} + \frac{1}{\mu_2}\frac{2-\rho_2}{2-2\rho_2}.$$

In this section, the response time of a range query is the total queuing delay on all the servers.

Detailed experiments are conducted on a Pentium IV 3-GHz system to study the execution time [29]. For G_1 over the Freeman–Scott–Teske (FST) curve, a single exponentiation operation in G_1 with 161 bits costs 1.1 ms, and the corresponding pairing operation costs 3.1 ms. Without loss of generality, let $N = 20$, $l = 5$. According to Table 4, the processing time of the CC is the tokens generation time, i.e., $1/\mu = (12l + 6) \times 1.1 = 72.6$ ms ≈ 0.073 s. If range query length is exponentially distributed with mean 2 Kbits and arrives according to a poisson process with rate 1query/10minute, i.e., $\lambda = 1/600$, and the queuing delay $T = \frac{1}{\mu-\lambda} = 0.073$ s and average queue length $L = \frac{\lambda}{\mu-\lambda} = 0.0012$.

Next, the processing time of the CS_2 depends on the query tokens verification and the searching time in CS_2's database. The computation time of a query token verification over one record is $10 \times 3.1 = 31$ ms. If the number of records in CS_2's database is $M = 100$, the CS_2's processing delay is 3.1 s, i.e. $1/\mu_2 = 3.1$ s. At last, querying indexed ciphertexts on the CS_1 is typically processed very fast, the processing time is only several milliseconds (e.g $1/\mu_1 = 10$ ms). Since $\mu_2 = max(\lambda_1)$, considering the extreme case $\lambda_1 = \mu_2$, then, $T_1 = 0.01$ s and $L_1 \approx 0$. Therefore, T, L, T_1, L_1 are very small, and few rang query tasks can be buffered in the queue CS_1 and CC. As a result, their queuing delay can be neglected.

From the above analysis, the processing time on the CC is much faster than the query arriving interval. Thus, $\lambda = \lambda_2 = 1/600$. Moreover, compared with the CS_2 queue, the service rate of the CS_1 and CC is much faster, i.e., $\mu_1 \gg \mu \gg \mu_2$. Therefore, a range query's average response time is mainly determined by the processing time of the CS_2. Consequently, the whole range query response time is distributed approximately as in the CS_2 queue, that is, as in an $M/D/1$ queue with poisson rate λ_2 and service rate $1/\mu_2$. Hence, the response time of a range query can be represented by T'_{tol}.

$$T'_{tol} = \frac{1}{\mu_2}\frac{2-\rho_2}{2-2\rho_2}.$$

If the total communication latency Δ among all network links is not negligible, the formula T'_{tol} should be adjusted to

$$T'_{tol} = \frac{1}{\mu_2}\frac{2-\rho_2}{2-2\rho_2} + \Delta.$$

In fact, the smart grid usually uses 3G (3rd Generation) or 4G (4th Generation) cellular network topology for cells data transmission. Data transmission rate is 60–240 Kbps, and distance converge depends upon the availability of cellular service [30]. Hence, the communications rate among the requesters, the CC and the CSs in our PaRQ scheme is assumed to be 240 Kbps. Here, $\lambda_2 = 1/600$ and $1/\mu_2 = 3.1$ s.

(1) If the communication latency Δ is negligible, i.e., $\Delta \approx 0$, we can see from Fig. 7 that the response time of a range query is increased with the number of the database records. Comparing the total response time of a range query with or without considering the queuing delay of the CC and CS_1 by using T_{tol} and T'_{tol}, respectively, Fig. 7 also shows that they are almost the same, which means that the queuing delay of the CS_1 and CC really can be neglected in response time calculation.

(2) If the communication latency Δ is not negligible, we should consider the communication overhead during the range query process. From the above analysis, if S sends a 2K-bit range query to the CC, the CC replies with two 1610-bit tokens. Then, S forwards these two tokens to the CS_2, and the CS_2 replies the satisfactory keys and indexes. If t is the average number of matched results and the size of the keys and indexes are 80 bits each, thus, their communication overhead is 160t bits. Finally, when S accesses the CS_1, the CS_1 replies the correct indexed ciphertexts. Usually, the size of the ciphertext is large. Let it be 1Mbits/packet. The communication overhead of the ciphertexts is 1tM bits. Hence, the system communication latency $\Delta \approx 0.022 + 4.26t$.

$$T'_{tol} = \frac{1}{\mu_2}\frac{2-\rho_2}{2-2\rho_2} + 0.022 + 4.26t.$$

Fig. 8 illustrates that both the number of data records in the CS_2's database and the number of replied ciphertexts from the CS_1 can augment the system response time.

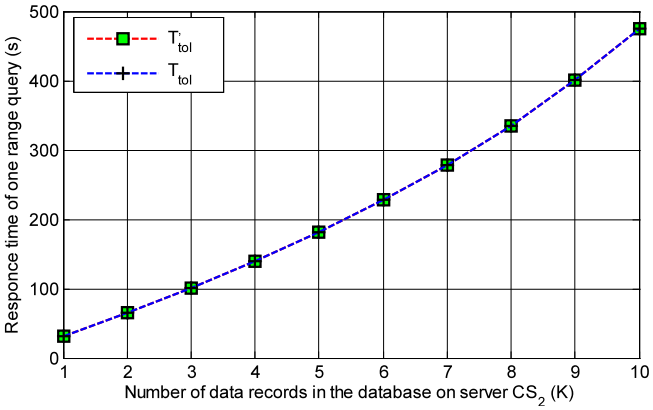

FIGURE 7. Response time in PaRQ scheme, $\Delta = 0$.

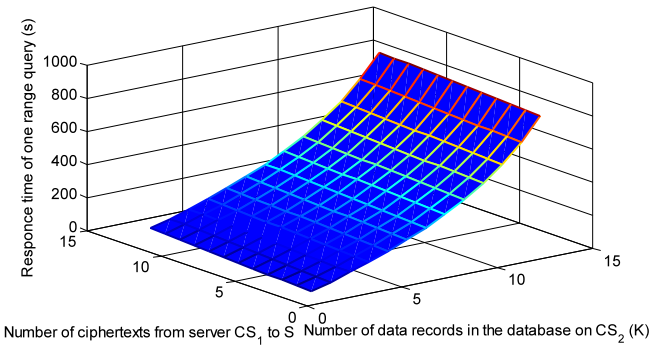

FIGURE 8. Response time in PaRQ scheme, $\Delta = 0.022 + 4.26t.c$

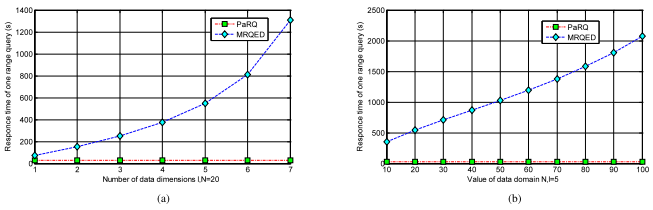

FIGURE 9. Comparison of the number of matched residential users. (a) $\lambda_2 = \lambda_2' = 1/600$, $1/\mu_2 = 3.1$ s, $1/\mu_2' = 0.6699$ l. (b) $\lambda_2 = \lambda_2' = 1/600$, $1/\mu_2 = 3.1$ s, $1/\mu_2' = 0.775 \times \log_2(N)$.

(3) Compared with the MRQED, Table 2 shows that the MRQED needs $5l \cdot l_{og}NC_p$ to verify a query on a ciphertext. In a database with the number of data records $M = 100$, the processing time in the database is $5l \cdot l_{og}N \times 3.1$ s. Similarly, in the MRQED, the encryption and decryption key generation time is much shorter than the processing time of data query in the database. Therefore, the range query response time of the MRQED is mainly determined by the query in database. Thus, the range query service in the MRQED can also be modeled by using an $M/D/1$ queue with poisson rate $\lambda_2' = 1/600$ and service rate $1/\mu_2' = 5l \cdot l_{og}N \times 3.1$. Fig. 9 illustrates the response time comparison between the PaRQ and the MRQED without communication delay, the range query arrival rates are $\lambda_2 = \lambda_2' = 1/600$. From Table 2 and Fig. 9, we can observe that in the MRQED, if the number of data records in the database is a constant, the response time of a range query increases with the domain N and the number of dimensions l, while the service time in our PaRQ is $1/\mu_2 = 3.1$ s. Thus, no matter how many dimensions of the data and how large the domain is, a single range query process time in the PaRQ remains $\frac{1}{\mu_2}\frac{2-\rho_2}{2-2\rho_2} = 3.1$ s, which is much less than that of the MRQED.

VIII. CONCLUSION

In this paper, we have proposed a privacy-preserving range query scheme, named PaRQ, for smart grid. An HVE based range query predicate is constructed to realize the range query on encrypted metering data. The PaRQ allows users to store their data on cloud servers in encrypted form, and range queries can be executed by using cloud server's computational capabilities. A requester with authorized query tokens can obtain the correct session keys to retrieve the metering data within specific query ranges. Security analysis demonstrates that the PaRQ can achieve data confidentiality and privacy and preserve query privacy. Performance evaluation shows that the PaRQ can significantly reduce computation and communication overhead, as well as response time. For our future work, we intend to enhance our PaRQ to support ranked range query with security and privacy preservation.

REFERENCES

[1] C. Lo and N. Ansari, "The progressive smart grid system from both power and communications aspects," *IEEE Commun. Surv. Tuts.*, vol. 14, no. 3, pp. 799–821, Mar. 2012.

[2] R. Zeng, Y. Jiang, C. Lin, and X. Shen, "Dependability analysis of control center networks in smart grid using stochastic petri nets," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 9, pp. 1721–1730, Sep. 2012.

[3] 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.

[4] C. Lo and N. Ansari, "Alleviating solar energy congestion in the distribution grid via smart metering communications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 9, pp. 1607–1620, Sep. 2012.

[5] C. Lo and N. Ansari, "Decentralized controls and communications for autonomous distribution networks in smart grid," *IEEE Trans. Parallel Distrib. Syst.*, vol. 4, no. 1, pp. 66–77, Mar. 2013.

[6] (Aug. 2010). The Smart Grid Interoperability Panel-Cyber Security Working Group, *Nistir 7628 Guidelines for Smart Grid Cyber Security: Smart Grid Cyber Security Strategy, Architecture, and High-Level Requirements*, China [Online]. Available: http://csrc.nist.gov/publications/nistir/ir7628/nistir-7628_vol1.pdf

[7] X. Liang, X. Li, R. Lu, X. Lin, and X. Shen, "UDP: Usage-based dynamic pricing with privacy preservation for smart grid," *IEEE Trans. Smart Grid*, vol. 4, no. 1, pp. 141–150, Mar. 2013.

[8] R. Yu, Y. Zhang, S. Gjessing, C. Yuen, S. Xie, and M. Guizani, "Cognitive radio based hierarchical communications infrastructure for smart grid," *IEEE Netw.*, vol. 25, no. 5, pp. 6–14, Sep./Oct. 2011.

[9] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[10] P. Sakarindr and N. Ansari, "Security services in group communications over wireless infrastructure, mobile Ad Hoc, and wireless sensor networks," *IEEE Wireless Commun.*, vol. 14, no. 5, pp. 8–20, Oct. 2007.

[11] X. Li, X. Liang, R. Lu, X. Shen, X. Lin, and H. Zhu, "Securing smart grid: Cyber attacks, countermeasures, and challenges," *IEEE Commun. Mag.*, vol. 50, no. 8, pp. 38–45, Aug. 2012.
[12] H. Li, R. Lu, L. Zhou, B. Yang, and X. Shen, "An efficient Merkle tree based authentication scheme for smart grid," *IEEE Syst. J.*, to be published.
[13] G. Acs and C. Castelluccia, "I have a DREAM! (DiffeRentially privatE smArt Metering)," in *Proc. 13th Int. Conf. Inf. Hiding*, 2011, pp. 118–132.
[14] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. Adv. Cryptol.*, 2004, pp. 506–522.
[15] M. Wen, R. Lu, J. Lei, H. Li, X. Liang, and X. Shen, "SESA: An efficient searchable encryption scheme for auction in emerging smart grid marketing," *Security Commun. Netw.*, to be published.
[16] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *Proc. Adv. Cryptol.*, 2011, pp. 578–595.
[17] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proc. ACM Int. Conf. Manag. Data SIGMOD*, 2004, pp. 563–574.
[18] B. Hore, S. Mehrotra, M. Canim, and M. Kantarcioglu, "Secure multidimensional range queries over outsourced data," *VLDB J.*, vol. 21, no. 3, pp. 333–358, 2012.
[19] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proc. TCC*, 2007, pp. 535–554.
[20] J. Park, "Efficient hidden vector encryption for conjunctive queries on encrypted data," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 10, pp. 1483–1497, Oct. 2011.
[21] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *Proc. Adv. Cryptol. EUROCRYPT*, 2008, pp. 146–162.
[22] V. Iovino and G. Persiano, "Hidden-vector encryption with groups of prime order," in *Proc. Pairing-Based Cryptograph. Pairing*, 2008, pp. 75–88.
[23] E. Shi, J. Bethencourt, T. Chan, D. Song, and A. Perrig, "Multi-dimensional range query over encrypted data," in *Proc. IEEE Symp. Security Privacy*, May 2007, pp. 350–364.
[24] D. Boneh and M. Franklin, "Identity-based encryption from the Weil pairing," in *Proc. Adv. Cryptol*, 2001, pp. 213–229.
[25] B. Libert and J. Quisquater, "The exact security of an identity based signature and its applications," Lab. d'Inform. l'École Polytech., Palaiseau, France, Tech. Rep. F-91128, 2004.
[26] J. Daemen, V. Rijmen, and A. Proposal, "Rijndael," in *Proc. 1st Adv. Encryption Standard Candidate Conf., NIST.*, 1998.
[27] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data Networks*, vol. 2. Upper Saddle River, NJ, USA: Prentice-Hall, 1992.
[28] J. Walrand, "A probabilistic look at networks of quasi-reversible queues," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 6, pp. 825–831, Nov. 1983.
[29] M. Scott, (2007). *Efficient Implementation of Cryptographic Pairings* [Online]. Available: http://www.pairing-conference.org/2007/invited/Scottslide.pdf
[30] P. Parikh, M. Kanabar, and T. Sidhu, "Opportunities and challenges of wireless communication technologies for smart grid applications," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2010, pp. 1–7.

MI WEN (M'10) received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2008. She is currently an Associate Professor with the College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai. Her current research interests include security in wireless sensor network and privacy preserving in smart grid.

RONGXING LU (S'09–M'11) received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China, in 2006, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2012. He is currently an Assistant Professor with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore. His current research interests include wireless network security, applied cryptography, and trusted computing.

KUAN ZHANG received the B.Sc. degree in electrical and computer engineering and the M.Sc. degree in computer science from Northeastern University, Shenyang, China, in 2009 and 2011, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His current research interests include packet forwarding, and security and privacy for mobile social networks.

JINGSHENG LEI is a Professor and the Dean of the College of Computer Science and Technology, Shanghai University of Electronic Power, Shanghaiÿ China. He received the B.S. degree in mathematics from Shanxi Normal University, Xian, China, in 1987, and the M.S. and Ph.D. degrees in computer science from Xinjiang University, Xinjiang, China, in 2000 and 2003, respectively. His current research interests include machine learning, data mining, pattern recognition and cloud computing. He has published more than 80 papers in international journals or conferences. He serves as an Editor-in-Chief of the *Journal of Computational Information Systems*. He is the member of the Artificial Intelligence and Pattern Recognition Technical Committee of the China Computer Federation and the Machine Learning Technical Committee of the Chinese Association of Artificial Intelligence.

XIAOHUI LIANG (S'10) received the B.Sc. degree in computer science and engineering and the M.Sc. degree in computer software and theory from Shanghai Jiao Tong University, Shanghai, China, in 2006 and 2009, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His current research interests include applied cryptography, and security and privacy issues for e-healthcare system, cloud computing, mobile social networks, and smart grid.

XUEMIN SHEN (M'97–SM'02–F'09) received the B.Sc. degree from Dalian Maritime University, Dalian, China, and the M.Sc. and Ph.D. degrees from Rutgers University, NJ, USA, in 1982, 1987, and 1990, respectively, all in electrical engineering. He is a Professor and University Research Chair with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. He was the Associate Chair for Graduate Studies from 2004 to 2008. His current research interests include resource management in interconnected wireless/wired networks, wireless network security, wireless body area networks, vehicular ad hoc, and sensor networks. He is the co-author or editor of six books, and has published more than 600 papers, and book chapters in wireless communications and networks, control, and filtering. He served as the Technical Program Committee Chair for IEEE VTC in 2010, the Symposia Chair for IEEE ICC in 2010, the Tutorial Chair for IEEE VTC in 2011, and IEEE ICC in 2008, the Technical Program Committee Chair for IEEE Globecom in 2007, the General Co-Chair for Chinacom in 2007 and QShine in 2006, the Chair for IEEE Communications Society Technical Committee on Wireless Communications, and P2P Communications and Networking. He serves/served as the Editor-in-Chief for the *IEEE Network*, *Peer-to-Peer Networking and Application*, and *IET Communications*, a Founding Area Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, an Associate Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, COMPUTER NETWORKS, and *ACM/Wireless Networks*, and the Guest Editor for the *IEEE JSAC*, *IEEE Wireless Communications*, the *IEEE Communications Magazine*, and *ACM Mobile Networks and Applications*. He received the Excellent Graduate Supervision Award in 2006, and the Outstanding Performance Award from the University of Waterloo in 2004, 2007, and 2010, the Premiers Research Excellence Award from the Province of Ontario, Canada, in 2003, and the Distinguished Performance Award from the Faculty of Engineering, University of Waterloo, in 2002 and 2007. He is a registered Professional Engineer of Ontario, Canada. He is a fellow of the Engineering Institute of Canada and the Canadian Academy of Engineering. He is a Distinguished Lecturer of IEEE Vehicular Technology Society and Communications Society. He has been a Guest Professor of Tsinghua University, Shanghai Jiao Tong University, Zhejiang University, Beijing Jiao Tong University, Northeast University.