ECQ: An Efficient Conjunctive Query Scheme over Encrypted Multidimensional Data in Smart Grid

Mi Wen†‡, Rongxing Lu‡, Jingshen Lei†, Xiaohui Liang‡, Hongwei Li§ and Xuemin (Sherman) Shen†
† School of Computer Engineering, Shanghai University of Electric Power, Shanghai, China 200090
‡ Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1
§ School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore

Index Terms—Conjunctive query, Data privacy, Smart grid, Multidimensional data.

I. INTRODUCTION

Smart grid has emerged as the next generation of power grid [1], [2]. By installing smart meters at users’ houses, the smart grid can collect real-time data about power consumption by residential users. The amount of data generated by smart meters and intelligent sensors in smart grid will experience explosive growth in the next few years. According to a recent report from SBI Energy [3], the volume of smart grid data that will have to be managed by utilities is going to surge from 10,780 terabytes (TB) in 2010 to over 75,200 TB in 2015 [4]. Consider that requesters, such as utility companies or marketing managers, are tasked with querying power system data by user types and/or dates, etc.[5]. How to query and mine this massive and heterogeneous power system raw data to support decision making and ensure reliability will be very critical for smart grid.

In smart grid, a short uploading interval is always desired to accurately reflect power usages. However, given the fact that the metering data of individual homes/factories is accumulated every 15 minutes, it is possible to infer the pattern of electricity consumption by individual users [6]. For example, the actions of the residents can be easily tracked by analyzing the smart meter data (gas, water, and electricity consumption). It is even possible to determine the presence/absence of residents or the number of people living in a household. In order to protect users’ privacy, the sensitive data should be stored in an encrypted form. Lu et al. [6] design an efficient and privacy-preserving aggregation scheme, named EPPA, by using the homomorphic paillier cryptosystem technique. Li et al. [7] propose an efficient merkle tree based authentication scheme for smart grid. However, these privacy preserving schemes cannot support queries on encrypted data.

Public key encryption with keyword search (PEKS) [8] is a widely studied approach to achieve querying on encrypted data. Nevertheless, most of the existing schemes (such as [9],[10]) about PEKS focus only on the keyword search technique, with little attention to both data and query privacy protection in one scheme. Baek et al. [11] argue that PEKS and data encryption schemes need to be treated as a single scheme to securely provide PEKS service. Qin et al.[12] propose an efficient encryption scheme with one-dimension keyword search (EPPKS) for cloud computing by combining the ideas of partial decipherment with the PEKS. However, it is not quite secure because if the server is untrusted, the partial decipherment will leak partial information of users’ data. Furthermore, the EPPKS cannot support conjunctive keyword searches on multiple dimensions. The reason is that power system data usually has multi-dimensional attributes. Taking into account that all these dimensions allow finer grained query, when a conjunctive query on multiple dimensions is posted, the EPPKS will have to process the query on every dimension separately.

To protect data privacy and save communication and computation overhead, in this paper, we propose an Efficient Conjunctive Query (ECQ) scheme in smart grid. This scheme supports conjunctive query on encrypted multi-dimensional data by giving conjunctive keywords on multiple dimensions. The main contributions of this paper are two-fold:

1) Firstly, we propose an efficient conjunctive query scheme, which considers both data and query privacy preservation when querying over the encrypted multi-dimensional data. Requesters can get matched results...
by giving conjunctive keywords on multiple dimensions. Compared with the EPPKS [12], the ECQ is more efficient in terms of user’s computation cost and total communication cost.

2) Secondly, we analyze the security strength and privacy preservation ability of the ECQ. The analysis results show that the ECQ can protect the data confidentiality and integrity, as well as data and query privacy. Even if the server is compromised, our proposed ECQ scheme can still protect data privacy without any information leakage, which is more secure than the EPPKS [12].

The remainder of this paper is structured as follows. In section II, we introduce our system model, security requirements and design goals. In section III, we recall some preliminary knowledge. The proposed scheme is described in section IV; followed by its security analysis and performance evaluation in section V. Finally, we draw our conclusion in section VI.

II. SYSTEM MODEL, SECURITY REQUIREMENTS AND DESIGN GOALS

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

A. System Model

In the system model, we mainly focus on how to query over encrypted multi-dimensional data for smart grid by using conjunctive keywords. Specifically, we consider a typical residential area, as shown in Fig.1, which comprises an operation center, a cloud server (CS), several local gateways (GA), a requester, and a large number of smart meters (SM) related to the corresponding residential users $U = \{U_1, U_2, \ldots, U_v\}$.

For data outsourcing, the power usage data of residential users will be stored on the cloud server forwarded by the GA. This is a promising approach to relieve the operation center from the burden of large amounts of data storage and maintenance, and executing computations and queries using the servers’ computational capabilities [13]. Requesters, such as market analysts and utility companies, will be able to query the cloud server to find useful information. The operation center can be considered as a trusted authority (TA), and can bootstrap the whole system. Specifically, the TA generates and distributes keys for residential users and authorized requesters.

B. Security Requirements

We define the security requirements for our efficient conjunctive query (ECQ) scheme, and will show the fulfillment of these requirements after presenting the design details.

- **Data Confidentiality and Integrity**: The power system data and queries should not be known and changed by malicious users or unauthorized users. That is, if an adversary $A$ maliciously modifies data on the CS, the power demand forecast and other policies will be mislead.

- **Data privacy**: As users usually prefer to keep their data from being exposed to others, including the CS, the most important concern is preserving data privacy. It means that only requesters with correct secret key can obtain the correct data when their query keywords are satisfied with those in the encrypted 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 trapdoors to protect the query privacy. Otherwise, if the query includes some sensitive information, such as “city hall”, then the CS could know the requester is querying some important locations’ metering data. Then, the requester or the query results could be traced or analyzed by the curious CS.

C. Design Goals

Under the aforementioned system model and security requirements, our design goal is to develop an efficient conjunctive query scheme for smart grid, achieving data security and efficient performance as follows.

- The proposed ECQ scheme should remain secure to meet the security requirements: the data confidentiality, integrity, and data and query privacy. Otherwise, the user-specific data leakage can lead to criminal targeting of homes.

- The proposed scheme should achieve conjunctive query efficiency over encrypted multidimensional data in terms of communication and computation overhead.

III. PRELIMINARIES

In this section, we will briefly describe the basic definition and properties of bilinear pairing and public-key encryption with keyword search.

A. Bilinear Pairing

Bilinear pairing is an important cryptographic primitive [14]. Let $G_1$ and $G_2$ be two cyclic multiplication groups of prime order $q$. Let $a$ and $b$ be elements of $\mathbb{Z}_q^*$. We assume that the discrete logarithm problem (DLP) in both $G_1$ and $G_2$
are hard. $g$ is a generator of $G_1$. A bilinear pairing is a map $e : G_1 \times G_1 \rightarrow G_2$ with the following properties:

1. Bilinear: $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in G_1$;
2. Non-degenerate: $e(g, h) \neq 1_{G_2}$ whenever $g, h \neq 1_{G_1}$;
3. Computable: There is an efficient algorithm to compute $e(g, h) \in G_2$ for all $(g, h) \in G_1^2$.

**Definition 1:** A bilinear parameter generator $Gen$ is a probabilistic algorithm that takes a security parameter $\kappa$ as input, and outputs a 5-tuple $(g, g, G_1, G_2, e)$.

### B. Public-key Encryption with Keyword Search

The concept of public key encryption with keyword search (PEKS) is proposed by Boneh et al. [8]. As defined, a PEKS scheme involves the following entities: sender, receiver, and a server. The workflow is as follows.

Firstly, a sender encrypts its message $m$ into a ciphertext $C$. Given a keyword $w$, the sender produces a searchable tag $C_w$ and sends the ciphertext $C$ and the tag $C_w$ to the server. Then, the server stores the ciphertext and the tags.

Secondly, a receiver selects a query $w$. It has the help of its secret key, the receiver computes a trapdoor $S_w$ and sends the trapdoor to the server. Given $S_w$ and $C_w$, the server tests weather $w = w'$ or not. If so, it returns the ciphertext $C$ to the receiver; otherwise it returns nothing.

### IV. THE PROPOSED ECQ SCHEME

In this section, we propose an efficient conjunctive query scheme on encrypted data for smart grid. It mainly consists of the following four phases: registration phase, data and tags encryption phase, conjunctive query phase, and data recovery phase. The ECQ incorporates and modifies the idea of keywords search approach [10] and the proxy re-encryption with keyword search [16]. The dataflow is shown in Fig. 2.

In system initialization phase, the TA can bootstrap the whole system and assign the key materials. TA first generates $(g, g_1, G_1, G_2, e)$ by running $Gen(1^\kappa)$, where $G_1$ and $G_2$ are two cyclic multiplicative groups of prime order $g$; and $g$ is a generator of group $G_1$. An identity based signature algorithm $Sig(\cdot)$ [15] is needed in our scheme. Furthermore, we need three hash functions $H_1 : \{0, 1\}^* \rightarrow G_1$, $H_2 : \{0, 1\}^* \rightarrow G_1$ and $H_3 : G_2 \rightarrow \{0, 1\}^n$ for some $n$. TA publishes the system parameters as $(g, g_1, G_1, G_2, e, H_1, H_2, H_3)$.

### A. Registration Phase

In our scheme, there are $n$ users $U = \{U_1, U_2, \ldots, U_n\}$ and $m$ requesters $SE = \{SE_1, SE_2, \ldots, SE_m\}$ in the system. The TA randomly chooses a master key $s \in Z_q^*$ and computes the corresponding public key $pk = g^s$. Thus, $(g^s, s)$ is the public/private key pair of the TA. When each $SE_j$ registers, the TA picks a random number $y_j \in Z_q$ and sets $pb_j = g^{y_j}$. $(pb_j, y_j)$ is $SE_j$’s public/private key pair. For each $U_i$, the TA assigns an ID-based key pair $(H_1(1D_{U_i}), H_1^*(1D_{U_i}))$ to it, denoted as $(vk_i, sk_i)$. Then, $TA$ also chooses parameters $g_1, b_0, b_1, \ldots, b_L \in G_1$, $h \in G_2$ and computes $h_1 = h^s$.

### B. Data and Tags Encryption Phase

Without loss of generality, we assume that there are $L$ dimensions in the power system data, and each dimension will be characterized by a keyword (e.g., user type, date etc.). Thus, $U_i$ selects a keyword set $W = \{w_1, w_2, \ldots, w_L\}$ to characterize its data and encrypts them as following steps.

1. $U_i$ denotes its identity-based public key as $A = vk_i$, and keeps the private key $sk_i$ secretly.
2. Let a $L$-dimensional power data be generated as $m_i = (pout_i, pin_i, po_i, T_{si}, \Delta)$, where $pin_i$ denotes the energy amount bought by $U_i$ from the grid; $pout_i$ is energy amount sold by $U_i$ to the grid; $po_i$ is $U_i$’s total payment; $T_{si}$ is the timestamp of the data and $\Delta$ is $U_i$’s other information. Then, $U_i$ selects a random $r_i \in Z_q^*$ and computes $C_{U_i} = H_3(e(g, H_2(A)^{r_i})) \oplus m_i$.
3. $U_i$ computes $B_i = (g^{y_i})^{r_i}$ and selects the elements $a, b, r \in Z_q$. Then, it constructs a $L$-degree polynomial:

$$f(x) = a(x - H_1(w_1)) (x - H_1(w_2)) \cdots (x - H_1(w_L)) + b = a_Lx^L + \cdots + a_1x + a_0. \tag{1}$$

In this way, $H_1(w_1), \ldots, H_1(w_L)$ are $L$ roots of the equation $f(x) - b = 0$. With the parameters $\{a_0, a_1, \ldots, a_L, b\}$, $U_i$ then computes a tag $F_i$:

$$\begin{align*}
C_0 &= h^{r^b}, \\
C_1 &= H_3(e(g_1, h)^{(a_0 + a_1 + \cdots + a_L)r_i}), \\
H_0 &= h_1^{a_0r_i} \cdots H_L = h_1^{a_Lr_i}; \\
P_B &= b_0^{a_0r_i} \cdots P_BL = b_L^{a_Lr_i}.
\end{align*} \tag{2}$$

We denote $F_i = (C_0, C_1, H_0, \ldots, H_L, P_B, \ldots P_BL, C_{U_i} = (B_i, F_i)$.
4. $U_i$ generates a signature $S_i = Sig(sk_i, C_{U_i}, C_{U_i})$ by using its secret key $sk_i$.
5. $U_i$ sends the encrypted data $K_i = (A, C_{U_i}, S_i, C_{U_i})$ to the GA. The GA forwards it to the CS.
6. The CS will store this information received from $U_i$ as a record $(U_i, K_i)$ in its database.

### C. Conjunctive Query Phase

TA computes two parameters $\alpha = g^{1/s}$, $\beta = g_1^{1/s}$ and secretly sends them to $SE_j$. If $SE_j$ needs to query the data
over $v$ dimensions. It will generate query $Q_j$ with keywords set $Q_j = \{w'_1, w'_2, ..., w'_i\}$. Then $SE_j$ generates a trapdoor $S_{w_i}$ and sends it to the CS. Without loss of generality, we assume, $\{w'_1, w'_2, ..., w'_i\}$ can be any subset of $\{w_1, w_2, ..., w_L\}$.

1) Trapdoor generation:

$SE_j$ picks a number $r \in Z_q$ and generates a trapdoor $S_{w_i}$ on the query $Q_j$ as:

$$
T_0 = g_1^{1/s}(g^{1/s}b_0)^r,
$$

$$
\ldots
$$

$$
T_L = g_1^{1/s}(g^{H_1(w_1)} + \ldots + H_1(w_i)^r/s)^r, \quad T_i = g_1^{1/s}(g^{H_1(w_i)^r}/sv_i b_i)^r,
$$

Let $S_{w_i} = (T_0, T_1, ... T_L; g^r, h_i^r)$. Then, $SE_j$ sends $S_{w_i}$ to the CS.

2) Query on the encrypted data:

For each encrypted data $K_i$ in CS’s database, the CS tests if $C_{U_i}$ satisfies the $SE_j$’s requirement $S_{w_i}$:

(1) Data verification:

- The CS verifies signature $S_i$ on data $(C_{U_i}, C_{U_i})$ with respect to $U_i$’s public key $A$.
- If the signature verification fails, the CS checks the next data in its database; else the CS goes on testing.

(2) Now, the CS computes some parameters as follows:

$$
A_1 = \prod_{i=0}^{L} e(T_i, H_i),
$$

$$
A_2 = e(g^{r}, C_0) = e(g^r, h^{r-b}),
$$

$$
A_3 = \prod_{i=0}^{L} e(PB_i, h_i^r).
$$

(3) The CS tests if $H_3(A_1/(A_2 A_3)) = C_1$. If so, $C_{U_i}$ will be stored in a data array $X[i]$; if not, $C_{U_i}$ will be rejected.

3) The CS sends results in data array $X[i]$ to $SE_j$.

D. Data Recovery Phase

Upon receiving data array $X[i]$ from the CS, the $SE_j$ can decrypt each $C_{U_i}$ in $X[i]$ as $m_i = C_{U_i} \oplus H_3(e(B_i, H_2(A))/y_i)$ by using its secret key $y_i$, otherwise, $C_{U_i}$ will be discarded. The decryption is as follows.

$$
C_{U_i} \oplus H_3(e(B_i, H_2(A))/y_i) = H_3(e(g, H_2(A)^r)) \oplus m_i \oplus H_3(e((g^y_i)^r, H_2(A))^{1/y_i})
$$

$$
= H_3(e(g, H_2(A)^r)) \oplus m_i \oplus H_3(e(g, H_2(A)^{r_i}))
$$

$$
= m_i
$$

V. PERFORMANCE ANALYSIS

A. Security Analysis

Since the idea of the ECQ is based on schemes [10] [16], the provable security of the ECQ can be easily reduced to them. Thus, the provable security analysis of the ECQ can reference to schemes [10] [16]. In this subsection, we just analyze the security properties of the proposed ECQ scheme.

- The confidentiality and integrity of the users’ data is achieved in the proposed ECQ scheme: In ECQ, each data is encrypted by its secret number $r_i$ as $C_{U_i} = H_3(e(g, H_2(A)^{r_i})) \oplus m_i$. Anyone who does not know $r_i$, can not recover $m_i$ from the ciphertext $C_{U_i}$. Furthermore, the data is signed by identity based signature scheme [15]. Since the signature $S_i = Sig(k_i, C_{U_i}, C_{U_i})$ is provably secure, the source authentication and data integrity can be guaranteed.

- The users’ data privacy and the requesters’ query privacy are also achieved in the proposed ECQ scheme: On one hand, as shown in Eq. (1) and Eq. (3), the keywords hash values are used to calculated the data tag and trapdoors and they are in the exponent of $g$. Anyone can not recover original keywords $w_i$ with the tag $C_{U_i}$ because of the one-way property of hash function.

On the other hand, when the CS tests the query on the encrypted data, it also cannot know anything about the keywords in the query, except for whether the test is successful or not. The correctness of the test $H_3(A_1/(A_2 A_3)) = C_1$ is as follows. Since

$$
A_1 = \prod_{i=0}^{L} e(g_1^{1/s}(g^{H_1(w_i)^r} + \ldots + H_1(w_i)^r/s)^r, h^{a_i + r})
$$

$$
= e(g_1, h)^{(a_0 + a_1 + \ldots + a_L)r - 1} e(g^{r}, h^{r-b})^{1/v} e(\sum_{i=0}^{L} a_i (H_1(w_i)^{r} + \ldots + H_1(w_i)^{r}), h^{a_i + r})
$$

$$
\prod_{i=0}^{L} e(b^{a_i + r - b}, h^r),
$$

(5)

if $H_1(w_i^1), ..., H_1(w_i^L)$ are the roots of equation $f(x) - b = 0$, which means that all the keywords in the trapdoor are the subset of the keywords in the tag, i.e. $\{w_1, w_2, ..., w_L\} \subseteq W$, then the CS can get the equation:

$$
1/v(\sum_{i=0}^{L} a_i H_1(w_i)^{r} + \ldots + H_1(w_i^L))
$$

$$
= 1/v(\sum_{i=0}^{L} a_i H_1(w_i)^{r} + \ldots + \sum_{i=0}^{L} a_i H_1(w_i^L))
$$

$$
= b.
$$

(6)

Therefore,

$$
A_1 = e(g_1, h)^{(a_0 + a_1 + \ldots + a_L)r - 1} e(g^{r}, h^{r-b})^{1/v} e(\sum_{i=0}^{L} b^{a_i + r - b}, h^r)
$$

$$
= e(g_1, h)^{(a_0 + a_1 + \ldots + a_L)r - 1} A_2 A_3.
$$

(7)

Notice that the keywords in the trapdoor can be listed in any order when the $SE_j$ computes the trapdoor. Substitute Eq.(7) into $H_3(A_1/(A_2 A_3))$, we have $C_1$. Thus, the test is successful, and the CS cannot know anything about the keywords in the query.

- The proposed ECQ can remain secure even when the server is compromised: If the CS is malicious, the user’s data
privacy will be leaked in EPPKS [12], because the partial decipherment key computed by the server will leak the partial information of the encrypted data. Whereas, in the proposed ECQ scheme the confidentiality of the encrypted data is always preserved even if the cloud server is malicious. From the above analysis, we can see that the ECQ can remain secure even when the server is compromised.

B. Performance Evaluation

In this subsection, we will evaluate the ECQ in terms of the computation and communication complexities.

Computation: In our ECQ, the computation tasks include pairing operations and exponentiation operations etc. For simplicity of description, the pairing and exponentiation operations are denoted as $C_p$ and $C_e$, respectively. For instance, there are $L$ dimensions in a power usage data $m$, and $m$ will be characterized by $L$ keywords in a keyword set $W$. $U_i$ will encrypt $m$ into a ciphertext $C_{U_i}$, and compute a tag $C_{W_i}$ on $W$. Therefore, it costs 4 pairings and $(2L + 4)$ exponentiation operations for $U_i$ to generate a data encryption $(A, C_{U_i}, S_i, C_{U_i})$, i.e., $4C_p + (2L + 4)C_e$. Then, $SE_j$ needs $(2L + 2)$ exponentiation operations to compute a trapdoor $S_w$. After receiving the trapdoor $S_w$ from $SE_j$, the CS needs 2 pairings to verify the signature, $(2L + 3)$ pairings and 3 exponentiation operations to test $SE_j$’s query. If there is a satisfactory result, $SE_j$ requires 1 pairing operation and 1 exponentiation operation to decrypt the ciphertext.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>COMPARISON OF COMPLEXITY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ECQ</td>
</tr>
<tr>
<td>user</td>
<td>$4C_p + (2L + 4)C_e$</td>
</tr>
<tr>
<td>CS</td>
<td>$(2L + 3)C_p + 3C_e$</td>
</tr>
<tr>
<td>Requester</td>
<td>$C_p + (2L + 3)C_e$</td>
</tr>
</tbody>
</table>

In comparison, for a $L$-dimensional data, in the traditional approach (e.g. EPPKS [12]) a user should generate $(L + 2)$ pairings and $(2L + 4)$ exponentiation operations to generate a data encryption with $L$ tags on $L$ dimensions. That totally costs $(L + 2)C_p + (2L + 4)C_e$. Since EPPKS [12] only can do 1 keyword query at a time, for a query with $L$ keywords, the requester needs to compute $L$ trapdoors, which needs $L$ exponentiation operations. And the sever needs to test $L$ times. Every time, the server needs 2 pairings to test whether a given tag matches a trapdoor or not. Thus, the sever totaly needs $2LC_p$ to test all of the trapdoors. If there is a satisfactory result, the sever needs $2C_p + 2C_e$ more computation overhead to get a partial decipherment. At last, it will take the requester $C_e$ to recover the real data.

Detailed experiments are also conducted on a Pentium IV 3-GHz system to study the execution time [17]. For $G_1$ over the FST curve, a single $C_e$ in $G_1$ with 161 bits costs 1.1ms, and the $C_p$ costs 3.1ms. From Table I and Fig.3-4 we can see that a user in the ECQ needs much less computation overhead than in the EPPKS [12]. Although the CS and requester need a little more computation overhead in the ECQ scheme than that in EPPKS [12], the ECQ needs less total computation overhead when the number of dimensions larger than 7. Thus, the ECQ is more suitable to the smart grid where there are more users and less requesters.

Communication: Most pairing-based cryptosystems need to work in a subgroup of the elliptic curve $E(F_q)$. If we represent elliptic curve points by using the point compression approach [18], the length of the elements in $G_1$ and $G_2$ will be roughly 161-bit and 1,024-bit, respectively. We assume that SHA-1 is used to compute the hash function, which yields a 160-bit output. The communications among the three entities of the proposed ECQ can be divided into three parts, user-to-CS, requester-to-CS and CS-to-requester communications.

We first consider the user-to-CS communication in the ECQ. In the data encryption phrase, users generate their encrypted data and tags, then they deliver them to the CS. The data report is in the form of $K_i = (A, C_{U_i}, S_i, C_{U_i})$, where, $A$ is a $G_1$ element; $C_{U_i}$ is a hash value; signature $S_i$ includes 2 $G_1$ elements [15]; $C_{U_i}$ includes $(2L + 1)G_1$ elements and 1 hash value; thus, the size of $K_i$ should be $161 * (2L + 1) + 160 \ast 2$ bits. In the requester-to-CS communication, each requester delivers a trapdoor $S_w = (T_0, T_1, ... T_{L-1}, q^*, h_i^*)$ to the CS, which includes $(L + 2)G_1$ elements, i.e., $161(L + 2)$ bits. In the CS-to-requester communication, the CS will reply a 160-bit ciphertext $C_{U_i}$ to $SE_j$.

In the EPPKS [12], the user sends a message $(C_m, C_{w_i}, i = 1, ... L)$ to the server, which includes a $G_1$ element and $L + 2$
hash elements. Its size is \((481 + 160L)\) bits. Then, a 160-bit long trapdoor will be sent from the requester to the server. \(L\) trapdoors are 160L bits. In the server-to-requester communication, the server replies \((C_m, C_p, C_{w_i}, i = 1, \ldots L)\) to the requester. Here, \(C_p\) is a \(G_2\) element, \(C_m\) includes a \(G_1\) element and 2 hash elements, \(C_{w_i}\) is a hash value. The size of the reply is \((160L + 1405)\) bits.

Table II and Figs. 5-6 show the comparison of communication overhead between the ECQ and EPPKS [12]. The ECQ needs much less communication overhead than the EPPKS [12], especially during CS-to-requester communication and total communication overhead.

**VI. CONCLUSION**

In this paper, we have developed an efficient conjunctive query scheme over encrypted multi-dimensional data in smart grid. In particular, it achieves both data and query privacy-preserving. Requesters, such as utility companies or marketing managers, can retrieve useful information by querying with conjunctive keywords. Security analysis demonstrates that our ECQ can achieve data confidentiality and integrity, data and query privacy. Simulation results show that our scheme can reduce the users’ computation cost and total communication cost. In future work, we intend to study ranked range queries on encrypted multidimensional data in smart grid.

**ACKNOWLEDGMENT**

This work is supported by the National Natural Science Foundation of China under Grant No.61073189, No.61103207, No.61272437 and No.61202369; the Project of Shanghai Local University Ability Construction under Grant No. 12510500700; and the Fundamental Research Funds for Chinese Central Universities under Grant ZYGX2011J059; and NSERC, Canada.

**TABLE II**

<table>
<thead>
<tr>
<th></th>
<th>ECQ</th>
<th>EPPKS [12]</th>
</tr>
</thead>
<tbody>
<tr>
<td>user-to-CS</td>
<td>161 + (2L + 1) + 320</td>
<td>481 + 160L</td>
</tr>
<tr>
<td>requester-to-CS</td>
<td>161(L + 2)</td>
<td>160L</td>
</tr>
<tr>
<td>CS-to-requester</td>
<td>160</td>
<td>160L + 1405</td>
</tr>
</tbody>
</table>

**REFERENCES**