A New Communication-Efficient Privacy-Preserving Range Query Scheme in Fog-Enhanced IoT

Rongxing Lu, *Senior Member, IEEE*

***Abstract*—Fog-enhanced Internet of Things (IoT) has received considerable attention in recent years, as fog devices deployed at the network edge can not only improve the performance of IoT applications but also enhance the security and privacy of IoT. In this paper, we present a new communication-efficient privacy-preserving range query scheme in Fog-enhanced IoT. With the proposed scheme, both the query range and individual IoT device's data can be privacy-preserved by using BGN homomorphic encryption technique. In addition, the proposed scheme employs a range query expression, decomposition, and composition technique to reorganize the range query, which can achieve $O(\sqrt{n})$ communication efficiency. Detailed security analysis shows that the proposed scheme is really a privacy-preserving range query scheme. Extensive experiments are conducted, and the results indicate that the proposed scheme is also efficient in terms of low range query generation cost and low communication overhead.**

***Index Terms*—Communication efficiency, fog-enhanced Internet of Things (IoT), privacy-preserving, range query.**

I. Introduction

As IT can bring great opportunities to change our daily lives, e.g., smart light bulbs illuminating smart homes, smart vehicles shuttling back, and forth in smart cities, Internet of Things (IoT) has received considerable attention in recent years. According to the Gartner's report in 2017, the number of connected IoT devices is around 8.4 billion currently and will possibly exceed 20 billion in 2020 [1]. The reason for IoT booming and changing our lives greatly is that a number of IoT devices in a specific IoT application form an interconnected network, which can not only collect but also share nearly real-time data for achieving better and intelligent decision. Despite the promising future of IoT, big data issues become challenging in some IoT applications [2], since huge volumes of data are generated for real-time analytics and decision. If all data need to be reported to the control center for processing, it will not only waste the scarce communication resources but also cause a long delay, unable to support real-time requirement in IoT domain. To address the challenges, one effort is to adopt more suitable real-time big data mining and machine learning techniques to IoT, and the other is to enhance the current IoT architecture with fog computing to deal with the big data issues [3].

The concept of fog computing has been widely discussed in IoT communities in the last five years, as it deploys fog devices at the network edge to enhance the IoT, i.e., solving problems or preprocessing parts of a problem at the network edge for better performance and quality of services [4]. The goal of fog computing is not to replace the cloud computing but to extend the computing and data processing capabilities to the network edge, so that the real-time and bandwidth challenges can be solved in the Fog-enhanced IoT. In addition, with the fog devices deployed at the network edge, the security and privacy can also be enhanced in IoT. For example, the fog device can not only *early* filter false injected data at the network edge but also protect each individual IoT device's data by adopting the privacy-preserving data aggregation techniques [2].

In this paper, we will focus ourselves on the privacy-preserving range query techniques in Fog-enhanced IoT. Assume that there are N IoT devices $\mathbf{D} = \{D_1, D_2, \ldots, D_N\}$, each $D_k \in \mathbf{D}$ has some sensed data w_k within the range $[1, n]$. Let us consider the following range query problem: a query user wants to know "how many IoT devices whose data are within the range $[L_B, U_B]$, where $1 \leq L_B \leq U_B \leq n$, are in $\mathbf{D}$, and what is the sum of their data?" For privacy reason, the query user does not want to disclose the query range, and each IoT device D_k also will not leak individual data w_k to others. In order to solve the above problem, a straightforward solution can be designed by using a homomorphic encryption $E()$, e.g., BGN [5] and Paillier encryption [6]. For example, when $n = 10$, $L_B = 4$, and $U_B = 7$, the query user first generates an array $A[1 \ldots 10] = \{0, 0, 0, 1, 1, 1, 1, 0, 0, 0\}$, encrypts A into a ciphertext, i.e., $E(A) = \{E(A[1]), E(A[2]), \ldots, E(A[10])\}$, and sends $E(A)$ to all IoT devices via the fog device at the network edge. According to its sensed data w_k, each IoT device D_k, picks up $E(A[w_k])$ from $E(A)$, uses the self blind technique to compute new ciphertexts $c_k \leftarrow \text{self-Blind}(E(A[w_k]))$, $s_k \leftarrow \text{self-Blind}(E(A[w_k])^{w_k})$, and reports (c_k, s_k) to the fog device. Note that, "self blind" is a useful property of $E()$, which can convert a valid ciphertext into another indistinguishable yet valid ciphertext for privacy preservation. The fog device runs the aggregation operation, i.e., $C = \prod_{D_k \in \mathbf{D}} c_k$, $S = \prod_{D_k \in \mathbf{D}} s_k$, and reports (C, S) to the query user. Finally, the query user can obtain the answers by decrypting (C, S). Obviously, the straightforward solution can achieve the desirable privacy requirements. However, when the range $[1, n]$ becomes large, e.g., $n = 10\,000$, the query is not communication efficient. Therefore, how to design a communication-efficient

Manuscript received June 1, 2018; revised August 23, 2018; accepted September 11, 2018. Date of publication September 19, 2018; date of current version May 8, 2019. This work was supported in part by NSERC Discovery Grants under Grant Rgpin 04009, in part by an NBIF Start-Up Grant under Grant Rif 2017-012, in part by HMF under Grant 2017 YS-04, and in part by URF under Grant Nf-2017-05 and Grant LMCRF-S-2018-03.

The author is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: rlu1@unb.ca).

Digital Object Identifier 10.1109/JIOT.2018.2871204

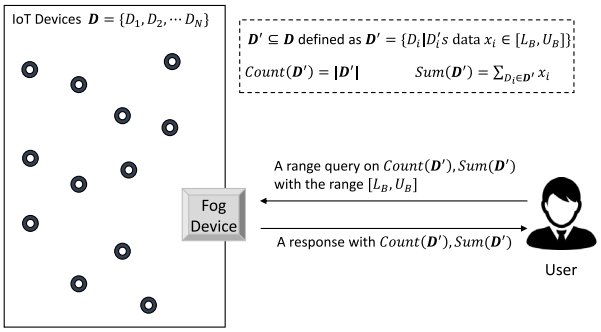

Fig. 1. System model under consideration.

privacy-preserving range query scheme in Fog-enhanced IoT becomes an interesting topic.

Aiming at the above problem, in this paper, we present a new communication efficient privacy-preserving range query scheme for Fog-enhanced IoT. The proposed scheme adopts a range query expression, decomposition, and composition technique to achieve $O(\sqrt{n})$ communication efficiency. Specifically, the main contributions of this paper are threefold.

1) In order to achieve $O(\sqrt{n})$ communication efficiency, we design a range query expression, decomposition, and composition technique, which can convert a range query from an array $A[1 \ldots n]$ into an $m \times m$ matrix R, where $n = m \times m$, and use five binary vectors, each of length m, to rebuild the matrix R.
2) Based on the range query expression, decomposition, and composition technique and BGN, we design our communication efficient privacy-preserving range query scheme for Fog-enhanced IoT.
3) We conduct extensive experiments to evaluate the performance of our proposed scheme, and the results show that it is indeed efficient in terms of low range query generation cost and low communication overhead.

The remainder of this paper is organized as follows. In Section II, we introduce our system model, security model, and design goal. Then, we describe some preliminaries in Section III. In Section IV, we present our privacy-preserving range query scheme, followed by security analysis and performance evaluation in Sections V and VI, respectively. Related work is discussed in Section VII. Finally, we draw our conclusion in Section VIII.

II. Models and Design Goal

In this section, we formalize our system model, security model, and identify our design goal.

A. System Model

In our system model, we consider a typical range query in Fog-enhanced IoT scenario, which mainly includes three types of entities, namely a set of IoT devices $\mathbf{D} = \{D_1, D_2, \ldots, D_N\}$, a fog device, and a query user, as shown in Fig. 1.

1) IoT Devices $\boldsymbol{D} = \{D_1, D_2, \ldots, D_N\}$: A set of IoT devices $\mathbf{D}$ of size N are deployed at some specific IoT domain. Each device $D_i \in \mathbf{D}$ is equipped with not only the sensing but also the communication capabilities, which enables D_i to periodically send the sensed data w_i to the fog device. For the simplicity and the clear description of our range query scheme later, we directly assume the value of w_i is an integer in the range of $[1, n]$, because even if the value of w_i is not an integer, e.g., $w_i = 3.782$, we still can easily transform the value by multiplying 1000 into $w_i = 3.782 \times 1000 = 3782$, though the upper bound n becomes relatively large.

2) Fog Device: A fog device is deployed at the network edge. Compared with the IoT devices, the fog device is more powerful in both computing and storing data. Therefore, the fog device can locally process the data collected from the IoT domain and respond to the IoT devices in an almost real-time way. In addition, the fog device can also be accessed by a query user, i.e., the fog device handles the range query from the query user, and responds the right results to the latter.

3) Query User: In our model, we consider a query user can directly launch some kinds of range queries to the fog device, and gain the desirable results from the fog device. For example, the query user will launch the following range query—"How many IoT devices, whose data are in the range $[L_B, U_B]$, where $1 \leq L_B \leq U_B \leq n$, are in the IoT domain? and what is the aggregated result of their data?" Assume the subset $\mathbf{D}' \subseteq \mathbf{D}$ is defined as

$$\mathbf{D}' = \left\{D_i | D_i's \text{ data } w_i \text{ is within } [L_B, U_B]\right\}. \tag{1}$$

Then, upon receiving the above range query, the fog device will return

$$\text{Count}\left(\mathbf{D}'\right) = \left|\mathbf{D}'\right| \text{ and } \text{Sum}\left(\mathbf{D}'\right) = \sum_{D_i \in \mathbf{D}'} w_i \tag{2}$$

to the query user as the respondent result.

Note that, the range query considered in our model is essentially a special privacy-preserving aggregation scheme with a private range given in a query, which can be applied in many Fog-enhanced IoT scenarios, e.g., smart grid, where a smart grid operator (user) who wishes to track the finer-grained electricity consumption of a neighborhood every 15 min, for scheduling and optimization purposes [2], [7].

B. Security Model

In our security model, we consider all entities are *honest-but-curious*, and there is no collusion among them. In other words, each entity will faithfully follow the protocol, however once certain conditions are satisfied, they will be curious about other entities' individual data. For example, the fog devices may be curious about each IoT device D_i's data w_i and the user's query range $[L_B, U_B]$; each IoT device may be curious about other IoT devices' data and the query range $[L_B, U_B]$; and the query user may be curious about each IoT device D_i's data w_i, in addition to Count($\mathbf{D}'$) and Sum($\mathbf{D}'$). Note that, an external adversary may launch other active attacks on data integrity and source authentication. However, since we focus on the communication-efficient privacy-preserving range query in this paper, those active attacks from an external adversary are beyond the scope of this paper and will be discussed in our future work, although it is not difficult to apply some mature

digital signature and message authentication code techniques to tackle these attacks.

C. Design Goal

Our design goal is to propose a communication-efficient privacy-preserving range query scheme in Fog-enhanced IoT to address the challenges mentioned in the above system model and security model. Specifically, the following two objectives should be attained.

1) Proposed Scheme Should Be Privacy-Preserving: In the proposed scheme, the query range $[L_B, U_B]$ should be privacy-preserving, i.e., no one, except the query user, can determine $[L_B, U_B]$. In addition, the elements of subset $\mathbf{D}'$ should also be privacy-preserving, i.e., no one can determine whether a specific IoT device belongs to $\mathbf{D}'$ or not, and only the query user can know Count($\mathbf{D}'$) and Sum($\mathbf{D}'$) after the range query.

2) Proposed Scheme Should Be Communication Efficient: In order to achieve the above privacy requirement in range queries, additional communication costs will be incurred in the range query, as we have discussed in Section I. Therefore, in the proposed scheme, we aim to make the query's communication efficient, i.e., achieving $O(\sqrt{n})$ communication efficiency.

III. Preliminaries

In this section, we briefly recall two basic building blocks used in our proposed scheme, namely the bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with composite order $\mathcal{N} = pq$ and the BGN homomorphic encryption [5].

A. Bilinear Pairing With Composite Order

Let p and q be two large primes of the same length, i.e., the bit-length $|p| = |q|$, and $\mathcal{N} = pq$. Two groups $(\mathbb{G}, \mathbb{G}_T)$ of the composite order $\mathcal{N}$ are called *bilinear map with composite order* if there is a computable mapping $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with the following three properties [5].

1) *Bilinearity:* $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G}^2$ and $a, b \in \mathbb{Z}_{\mathcal{N}}$.
2) *Nondegeneracy:* There exists $g \in \mathbb{G}$, such that $e(g, g)$ is with the order $\mathcal{N}$ in $\mathbb{G}_T$.
3) *Computability:* There exists an efficient algorithm to compute $e(g, h) \in \mathbb{G}_T$ for all $(g, h) \in \mathbb{G}$.

Definition 1 (Composite Bilinear Parameter Generator): A composite bilinear parameter generator $\mathcal{CG}en$ is a probabilistic algorithm that takes a security parameter κ as input, and outputs a five-tuple $(\mathcal{N}, g, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathcal{N} = pq$ and p, q are two κ-bit prime numbers, $\mathbb{G}$, $\mathbb{G}_T$ are two groups with order $\mathcal{N}$, $g \in \mathbb{G}$ is a generator, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a nondegenerated and efficiently computable bilinear map.

Let $\mathbf{g}$ be a generator of $\mathbb{G}$, then $g = \mathbf{g}^q \in \mathbb{G}$ can generate the subgroup $\mathbb{G}_p = \{g^0, g^1, \ldots, g^{p-1}\}$ of order p, and $g' = \mathbf{g}^p \in \mathbb{G}$ can generate the subgroup $\mathbb{G}_q = \{g'^0, g'^1, \ldots, g'^{q-1}\}$ of order q in $\mathbb{G}$. The subgroup decision (SGD) Problem in $\mathbb{G}$ is stated as follows [5]: given a tuple $(e, \mathbb{G}, \mathbb{G}_T, \mathcal{N}, h)$, where the element h is drawn randomly from either $\mathbb{G}$ or subgroup $\mathbb{G}_q$, decide whether $h \in \mathbb{G}_q$ or not. When we assume that the SGD problem is hard, the security of the BGN homomorphic encryption can be ensured [5].

B. BGN Homomorphic Encryption

The BGN is a famous homomorphic encryption [5], which mainly consists of three algorithms: 1) key generation; 2) encryption; and 3) decryption.

1) Key Generation: Given the security parameter κ, composite bilinear parameters $(\mathcal{N}, g, \mathbb{G}, \mathbb{G}_T, e)$ are generated by $\mathcal{CG}en(\kappa)$, where $\mathcal{N} = pq$ and p, q are two κ-bit prime numbers, and $g \in \mathbb{G}$ is a generator of order $\mathcal{N}$. Set $h = g^q$, then h is a random generator of the subgroup of $\mathbb{G}$ of order p. The public key is $pk = (\mathcal{N}, \mathbb{G}, \mathbb{G}_T, e, g, h)$, and the corresponding private key is $sk = p$.

2) Encryption: We assume the message space consists of integers in the set $\mathbb{S} = \{0, 1, \ldots, \Delta\}$, the size of set $\mathbb{S} = \{0, 1, \ldots, \Delta\}$ is application-oriented and much smaller than q, i.e., $\Delta \ll q$. To encrypt a message $m \in \mathbb{S}$, we choose a random number $r \in \mathbb{Z}_{\mathcal{N}}$, and compute the ciphertext $c = E(m, r) = g^m h^r \in \mathbb{G}$.

3) Decryption: Given the ciphertext $c = E(m, r) = g^m h^r \in \mathbb{G}$, the corresponding message can be recovered by the private key p. Observe that $c^p = (g^m h^r)^p = (g^p)^m$, we can set $\hat{g} = g^p$. Then, to recover m, it suffices to compute the discrete log of c^p base $\hat{g}$. Since $0 \le m \le \Delta$, the expected time is around $O(\sqrt{\Delta})$ when using the Pollard's lambda method [8, p. 128].

The famous BGN encryption has the property of *self-blindness*, i.e., given $E(m, r) \in \mathbb{G}$, we have $E(m, r + r') \leftarrow E(m, r) \cdot h^{r'}$ is also a valid ciphertext of m. In addition, BGN also enjoys the following homomorphic properties.

1) *Addition in* $\mathbb{G}$*:* Given $E(m_1, r_1) \in \mathbb{G}$ and $E(m_2, r_2) \in \mathbb{G}$, we have $E(m_1, r_1) \cdot E(m_2, r_2) = E(m_1 + m_2, r_1 + r_2) \in \mathbb{G}$. For simplicity, we omit the random items, and we have $E(m_1) \cdot E(m_2) = E(m_1 + m_2)$.
2) *Multiplication in* $\mathbb{G}$*:* Given $E(m_1, r_1) \in \mathbb{G}$ and $m_2 \in \mathbb{S}$, we have $E(m_1, r_1)^{m_2} = E(m_1 \cdot m_2, r_1 \cdot m_2) \in G$. For simplicity, we have $E(m_1)^{m_2} = E(m_1 \cdot m_2)$.
3) *Multiplication From* $\mathbb{G}$ *to* $\mathbb{G}_T$*:* Given $E(m_1), E(m_2) \in \mathbb{G}$, we have $e(E(m_1), E(m_2)) = E_T(m_1 \cdot m_2) \in \mathbb{G}_T$, where $E_T(\cdot)$ denotes a ciphertext in $\mathbb{G}_T$.
4) *Addition in* $\mathbb{G}_T$*:* Given $E_T(m_1), E_T(m_2) \in \mathbb{G}_T$, we have $E_T(m_1) \cdot E_T(m_2) = E_T(m_1 + m_2)$.
5) *Multiplication in* $\mathbb{G}_T$*:* Given $E_T(m_1) \in \mathbb{G}_T$ and $m_2 \in \mathbb{S}$, we have $E_T(m_1)^{m_2} = E_T(m_1 \cdot m_2)$.

IV. Our Proposed Scheme

In this section, we will present our new communication-efficient privacy-preserving range query scheme in Fog-enhanced IoT, which mainly consists of five parts: 1) query user key generation; 2) range query generation at query user; 3) query response at IoT devices; 4) response aggregation at fog device; and 5) response recovery at query user. Since our goal is to achieve communication efficiency in privacy-preserving range query, before delving into the details of our proposed scheme, we first introduce a very elegant range query expression, decomposition, and composition technique

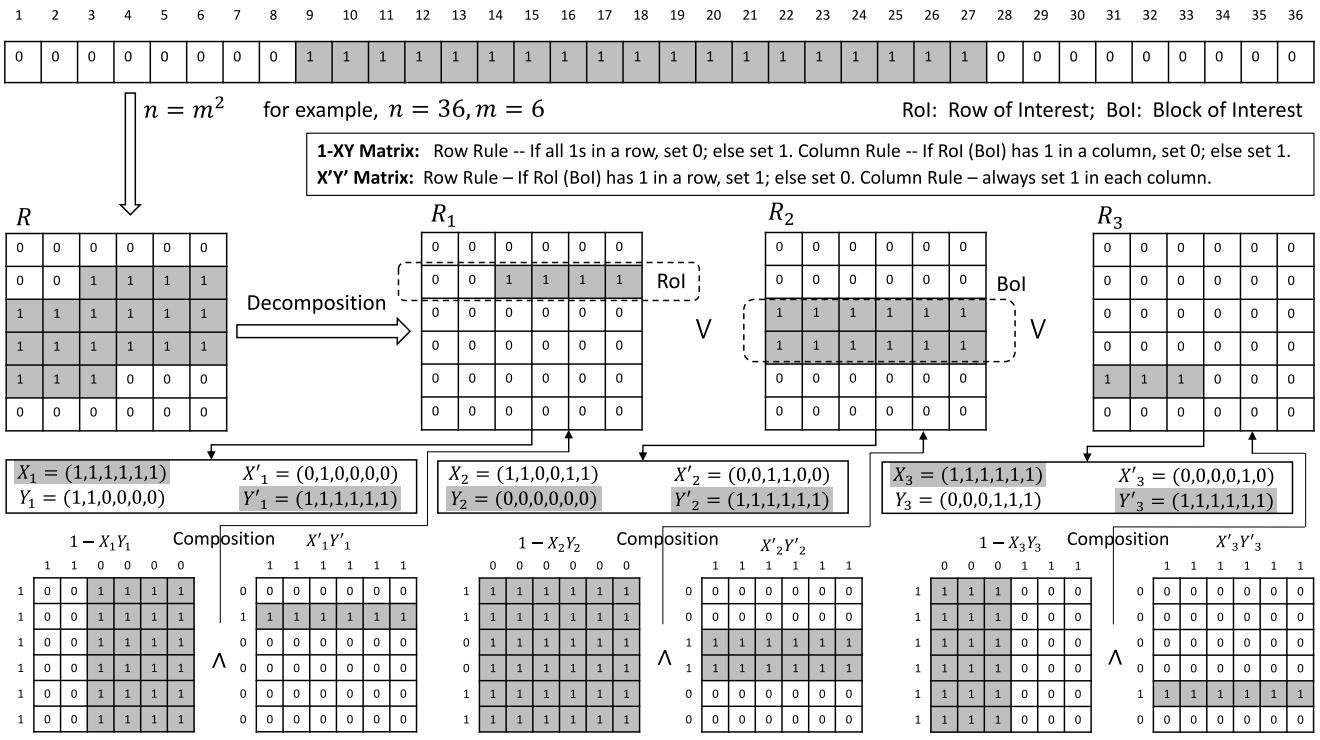

Fig. 2. Example for range query expression, decomposition, and composition.

for achieving $O(\sqrt{n})$ communication efficiency, which should be considered as the core contribution of this paper.

A. Range Query Expression, Decomposition, and Composition

In order to achieve $O(\sqrt{n})$ communication efficiency, we need to reorganize the range query by the following expression, decomposition, and composition technique.

1) Range Query Expression: Let L_B and U_B, respectively, be the lower bound and the upper bound of a range query $[L_B, U_B]$, we will have $1 \le L_B \le U_B \le n$. For the range query $[L_B, U_B]$, we can originally use an array $A[1 \ldots n]$ to represent it, i.e., for $1 \le k \le n$

$$A[k] = \begin{cases} 1, & L_B \le k \le U_B \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

However, when n is large, directly sending $A[1 \ldots n]$ for privacy preservation is not communication efficient, as discussed in Section I. For communication efficiency, we need to reorganize the range query $A[1 \ldots n]$ into an $m \times m$ matrix R. Without loss of generality, we can assume $n = m^2$, because if n is not a square number, we can easily increase n so that the condition $n = m^2$ holds. In order to map the range query from the array $A[1 \ldots n]$ to the matrix R, we have the following mapping relationship for each matrix element $R(i, j)$, where $1 \le i, j \le m$:

$$R(i,j) = \begin{cases} 1, & L_B \le k = (i-1) \times m + j \le U_B \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

When $n = 36$ and $m = 6$, Fig. 2 shows an example on how to transform a range query $[L_B = 9, U_B = 27]$ from an array A to a matrix R.

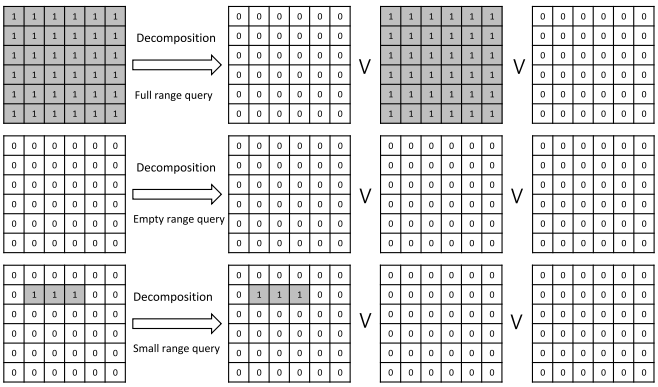

Fig. 3. Decomposition rule for special range query cases.

2) Range Query Decomposition: In matrix R, we define the row of interest (RoI) as a row in which not all elements are 0s or 1s, e.g., the row 2 and row 5 in R in Fig. 2; and the block of interest (BoI) as a set of continuous rows in which all elements are 1s, e.g., the block formed by the row 3 and row 4 in R in Fig. 2.

Decomposition Rule for R: In order for range query decomposition, we first break down R into three matrices R_1, R_2, and R_3, such that $R = R_1 \bigvee R_2 \bigvee R_3$, the matrix R_1 includes at most one RoI, the matrix R_2 includes at most one BoI, and the matrix R_3 also includes at most one RoI. For example, in Fig. 2, R_1 includes the RoI (row 2), R_2 includes the BoI (row 3, row 4), and R_3 includes the RoI (row 5). Note that, for some special range queries, such as the full range query, empty range query, and short range query, as shown in Fig. 3,

we can also apply the decomposition rule to decompose R into R_1, R_2, and R_3.

Decomposition Rule for R_w (w = 1, 2, 3): For achieving the communication-efficient range query, we need to further discompose R_w into two matrices $R_{1-X_wY_w}$ and $R_{X'_wY'_w}$ such that $R_w = R_{1-X_wY_w} \bigwedge R_{X'_wY'_w}$. The decomposition rule for R_w is as follows.

1) *Generate $1 - X_wY_w$ Matrix:* Set a row vector $X_w = (x_{w1}, x_{w2}, \ldots, x_{wm})$ of the length m with the following row rule: if the row i in the matrix R_w are all 1s, set $x_{wi} = 0$; and set $x_{wi} = 1$ otherwise. Set a column vector $Y_w = (y_{w1}, y_{w2}, \ldots, y_{wm})$ of the length m with the following column rule: if RoI (BoI) in matrix R_w has an element 1 in column j, that is, the column j in R_w are not all 0s, set $y_{wj} = 0$; and set $y_{wj} = 1$ otherwise. Then, each element $R_{1-X_wY_w}(i, j)$ in matrix $R_{1-X_wY_w}$ can be generated by vectors X_w, Y_w as

$$R_{1-X_wY_w}(i, j) = 1 - x_{wi}y_{wj}. \tag{5}$$

2) *Generate $X'_wY'_w$ Matrix:* Set a row vector $X'_w = (x'_{w1}, x'_{w2}, \ldots, x'_{wm})$ of the length m with the following row rule: if RoI (BoI) in matrix R_w has an element 1 in row i, that is, the row i in R_w are not all 0s, set $x'_{wi} = 1$; and set $x'_{wi} = 0$ otherwise. Set a column vector $Y'_w = (y'_{w1}, y'_{w2}, \ldots, y'_{wm})$ of the length m with the following column rule: always set each $y'_{wj} = 1$. Then, each element $R_{X'_wY'_w}(i, j)$ in matrix $R_{X'_wY'_w}$ can be generated by vectors X'_w, Y'_w as

$$R_{X'_wY'_w}(i, j) = x'_{wi}y'_{wj}. \tag{6}$$

By checking the example in Fig. 2, we can easily see that the above decomposition rules are correct. From matrices $R_{1-X_wY_w}$ and $R_{X'_wY'_w}$, we can recover $R_w = R_{1-X_wY_w} \bigwedge R_{X'_wY'_w}$ for $w = 1, 2, 3$. From matrices R_1, R_2, and R_3, we can recover $R = R_1 \bigvee R_2 \bigvee R_3$.

3) Range Query Composition: Since the matrix $R_{1-X_wY_w}$ is generated by two vectors (X_w, Y_w), and the matrix $R_{X'_wY'_w}$ is generated by two vectors (X'_w, Y'_w) for $w = 1, 2, 3$, we can see the matrix R can be recovered by 12 vectors $(X_1, Y_1, X'_1, Y'_1, X_2, Y_2, X'_2, Y'_2, X_3, Y_3, X'_3, Y'_3)$. For instance, each element $R(i, j)$ in the matrix R, for $1 \le i, j \le m$, can be computed as follows:

$$\begin{aligned} R(i,j) &= R_1(i,j) \vee R_2(i,j) \vee R_3(i,j) \\ &= \vee_{w=1}^{3}\left(R_{1-X_wY_w}(i,j) \wedge R_{X'_wY'_w}(i,j)\right) \\ &= \vee_{w=1}^{3}\left(\left(1 - x_{wi}y_{wj}\right) \wedge x'_{wi}y'_{wj}\right) \\ &\xrightarrow{\text{because bit AND, OR operations}} \\ &= \sum_{w=1}^{3}\left(1 - x_{wi}y_{wj}\right) \cdot x'_{wi}y'_{wj} \\ &= \begin{cases} 1, & \text{within the query range} \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{7}$$

Because each vector is of length $m = \sqrt{n}$, we just need to store these vectors with cost $O(12 \cdot m) = O(12 \cdot \sqrt{n}) = O(\sqrt{n})$ to reconstruct the rang query. Further observe the 12 vectors, we can see all elements in vectors $(X_1, X_3, Y'_1, Y'_2, Y'_3)$ are 1s from the decomposition rule; and when the matrix R_2 includes one BoI, all elements in the vector Y_2 are all 0s. Therefore, it is possible for us not to store vectors $(X_1, X_3, Y_2, Y'_1, Y'_2, Y'_3)$, and we still can recover the matrix R only from vectors $(Y_1, X'_1, X_2, X'_2, Y_3, X'_3)$. That is, (7) can be rewritten as

$$\begin{aligned} R(i,j) &= \sum_{w=1}^{3}\left(1 - x_{wi}y_{wj}\right) \cdot x'_{wi}y'_{wj} \\ &\xrightarrow{\text{because } y'_{wj}=1, x_{1i}=1, x_{3i}=1, y_{2j}=0} \\ &= \left(1 - y_{1j}\right) \cdot x'_{1i} + (1 - x_{2i} \cdot 0) \cdot x'_{2i} + \left(1 - y_{3j}\right) \cdot x'_{3i} \\ &= \left(1 - y_{1j}\right) \cdot x'_{1i} + x'_{2i} + \left(1 - y_{3j}\right) \cdot x'_{3i} \\ &= \begin{cases} 1, & \text{within the query range} \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{8}$$

Special Cases: When the matrix R_2 does not include any BoI, e.g., the two special range queries in Fig. 3, Y_2 becomes all 1s. However, in that case, the vector X'_2 becomes all 0s, then we can still use the same formula in (7) to compute $R(i, j)$, because

$$\begin{aligned} R(i,j) &= \sum_{w=1}^{3}\left(1 - x_{wi}y_{wj}\right) \cdot x'_{wi}y'_{wj} \\ &\xrightarrow{\text{because } y'_{wj}=1, x_{1i}=1, x_{3i}=1, y_{2j}=1, x'_{2i}=0} \\ &= \left(1 - y_{1j}\right) \cdot x'_{1i} + (1 - x_{2i} \cdot 1) \cdot 0 + \left(1 - y_{3j}\right) \cdot x'_{3i} \\ &= \left(1 - y_{1j}\right) \cdot x'_{1i} + 0 + \left(1 - y_{3j}\right) \cdot x'_{3i} \\ &= \left(1 - y_{1j}\right) \cdot x'_{1i} + x'_{2i} + \left(1 - y_{3j}\right) \cdot x'_{3i}. \end{aligned} \tag{9}$$

Based on the above analysis, it is surprising to see that the vector X_2 is also not needed to recover the matrix R. Therefore, if the query user wants to launch a range query, he/she just needs to send $(Y_1, X'_1, X'_2, Y_3, X'_3)$ as the query request. Then, the communication cost is $O(5 \cdot m) = O(5 \cdot \sqrt{n}) = O(\sqrt{n})$.

B. Description of Our Proposed Scheme

In this section, based on the above nice result, we present our communication-efficient privacy-preserving range query scheme in Fog-enhanced IoT.

1) Query User Key Generation: Given the security parameter κ, the query user generates the BGN public key $pk = (\mathcal{N}, \mathbb{G}, \mathbb{G}_T, e, g, h)$, and the corresponding private key $sk = p$. Then, the query user keeps the private key sk secretly, and publishes the public key pk. In order to fit for the query on $\text{Sum}(\mathbf{D}') = \sum_{D_i \in \mathbf{D}'} w_i$, the query user sets the message space $\mathbb{S} = \{0, 1, \ldots, \Delta\}$, where $\Delta \ge n \cdot N$.

2) Range Query Generation at Query User: Assume that the query user launches the following range query—"How many IoT devices, whose data are in the range $[L_B, U_B]$, where $1 \le L_B \le U_B \le n = m^2$, are in the IoT domain? and what is the aggregated result of their data?" In other words, the query user wants to query $\text{Count}(\mathbf{D}') = |\mathbf{D}'|$ and $\text{Sum}(\mathbf{D}') = \sum_{D_i \in \mathbf{D}'} w_i$, where $\mathbf{D}' = \{D_i | D'_i\text{s data } w_i \text{ is within } [L_B, U_B]\}$. In order to achieve the privacy preservation in the above range query, the query user runs the following steps.

Step 1: The query user applies the range query expression technique discussed above to convert the range query into a

matrix R of size $m \times m$, where each element

$$R(i,j) = \begin{cases} 1, & L_B \le k = (i-1) \times m + j \le U_B \\ 0, & \text{otherwise.} \end{cases}$$

Step 2: Because each element $R(i,j)$ can be computed by

$$R(i,j) = \left(1 - y_{1j}\right) \cdot x'_{1i} + x'_{2i} + \left(1 - y_{3j}\right) \cdot x'_{3i} \tag{10}$$

the query user applies the decomposition rules discussed above to prepare five binary vectors $Y_1 = (y_{11}, y_{12}, \ldots, y_{1m})$, $X'_1 = (x'_{11}, x'_{12}, \ldots, x'_{1m})$, $X'_2 = (x'_{21}, x'_{22}, \ldots, x'_{2m})$, $Y_3 = (y_{31}, y_{32}, \ldots, y_{3m})$, and $X'_3 = (x'_{31}, x'_{32}, \ldots, x'_{3m})$, where each element in all vectors is either 0 or 1. For the computation efficiency, the query user further computes two vectors $(\bar{Y}_1, \bar{Y}_3)$ from (Y_1, Y_3), where

$$\bar{Y}_1 = (\bar{y}_{11} = 1 - y_{11}, \bar{y}_{12} = 1 - y_{12}, \ldots, \bar{y}_{1m} = 1 - y_{1m})$$
$$\bar{Y}_3 = (\bar{y}_{31} = 1 - y_{31}, \bar{y}_{32} = 1 - y_{32}, \ldots, \bar{y}_{3m} = 1 - y_{3m}).$$

In this way, (10) can be rewritten as

$$R(i,j) = \bar{y}_{1j} \cdot x'_{1i} + x'_{2i} + \bar{y}_{3j} \cdot x'_{3i}. \tag{11}$$

Step 3: The query user uses the BGN to encrypt the five vectors $(\bar{Y}_1, X'_1, X'_2, \bar{Y}_3, X'_3)$ as

$$\begin{cases} E\left(\bar{Y}_1\right) = (E(\bar{y}_{11}), E(\bar{y}_{12}), \ldots, E(\bar{y}_{1m})) \\ E\left(X'_1\right) = \left(E\left(x'_{11}\right), E\left(x'_{12}\right), \ldots, E\left(x'_{1m}\right)\right) \\ E\left(X'_2\right) = \left(E\left(x'_{21}\right), E\left(x'_{22}\right), \ldots, E\left(x'_{2m}\right)\right) \\ E\left(\bar{Y}_3\right) = (E(\bar{y}_{31}), E(\bar{y}_{32}), \ldots, E(\bar{y}_{3m})) \\ E\left(X'_3\right) = \left(E\left(x'_{31}\right), E\left(x'_{32}\right), \ldots, E\left(x'_{3m}\right)\right) \end{cases} \tag{12}$$

and then sends $(E(\bar{Y}_1), E(X'_1), E(X'_2), E(\bar{Y}_3), E(X'_3))$ as the query to all IoT devices $\mathbf{D} = \{D_1, D_2, \ldots, D_N\}$ via the fog device.

Note that, for achieving the computation efficiency, the query user can precompute a pile of ciphertexts of 1 and 0, i.e., $\{E(0), \ldots, E(0), E(1), \ldots, E(1)\}$, and uses them to form the query $(E(\bar{Y}_1), E(X'_1), E(X'_2), E(\bar{Y}_3), E(X'_3))$ when launching a query, which can accelerate the query generation.

3) Query Response at IoT Devices: For each IoT device $D_k \in \mathbf{D}$ with sensed data $w_k \in [1, n]$, after receiving $(E(\bar{Y}_1), E(X'_1), E(X'_2), E(\bar{Y}_3), E(X'_3))$, it performs the following steps.

Step 1: D_k converts the sensed data w_k into (i,j) such that

$$w_k = (i-1) \times m + j, \text{ with } 1 \le i, j \le m.$$

Step 2: D_k picks up $E(\bar{y}_{1j})$, $E(x'_{1i})$, $E(x'_{2i})$, $E(\bar{y}_{3j})$, $E(x'_{3i})$, chooses two random numbers $r_{k_1}, r_{k_2} \in \mathbb{Z}_{\mathcal{N}}$, and computes

$$\begin{aligned} c_k &= e\left(E\left(\bar{y}_{1j}\right), E\left(x'_{1i}\right)\right) \cdot e\left(E\left(x'_{2i}\right), g\right) \\ &\quad \times e\left(E\left(\bar{y}_{3j}\right), E\left(x'_{3i}\right)\right) \cdot e(g,h)^{r_{k_1}} \\ &= E_T\left(\bar{y}_{1j} \cdot x'_{1i} + x'_{2i} + \bar{y}_{3j} \cdot x'_{3i}\right) \\ &= E_T(R_k(i,j)) \end{aligned} \tag{13}$$

$$\begin{aligned} s_k &= c_k^{w_k} \cdot e(g,h)^{r_{k_2}} \\ &= E_T(R_k(i,j))^{w_k} \cdot e(g,h)^{r_{k_2}} \\ &= E_T(R_k(i,j) \cdot w_k). \end{aligned} \tag{14}$$

Note that, for achieving a better efficiency, $e(g,h)^{r_{k_1}}$, $e(g,h)^{r_{k_2}}$ can be precomputed by the IoT device, and $e(E(x'_{2i}), g)$ can also be directly sent by the query user.

Step 3: D_k forwards the result (c_k, s_k) to the fog device.

4) Response Aggregation at Fog Device: After receiving all (c_k, s_k) from all $D_k \in \mathbf{D}$, the fog device computes

$$C = \prod_{D_k \in \mathbf{D}} c_k = E_T\left(\sum_{D_k \in \mathbf{D}} R_k(i,j)\right) \tag{15}$$

$$S = \prod_{D_k \in \mathbf{D}} s_k = E_T\left(\sum_{D_k \in \mathbf{D}} R_k(i,j) \cdot w_k\right) \tag{16}$$

and returns (C, S) as the response to the query user.

5) Response Recovery at Query User: Upon receiving (C, S), the query user uses the BGN private key $sk = p$ to recover the query results as

$$C \xrightarrow{\text{dec}} \text{Count}(\mathbf{D}') = \left|\mathbf{D}'\right| = \sum_{D_k \in \mathbf{D}} R_k(i,j) \tag{17}$$

$$S \xrightarrow{\text{dec}} \text{Sum}(\mathbf{D}') = \sum_{D_i \in \mathbf{D}'} w_i = \sum_{D_k \in \mathbf{D}} R_k(i,j) \cdot w_k. \tag{18}$$

The correctness of the above results are as follows:

$$\sum_{D_k \in \mathbf{D}} R_k(i,j) = \sum_{D_k \in \mathbf{D}'} 1 + \sum_{D_k \notin \mathbf{D}'} 0 = \left|\mathbf{D}'\right|$$

$$\sum_{D_k \in \mathbf{D}} R_k(i,j) \cdot w_k = \sum_{D_k \in \mathbf{D}'} 1 \cdot w_k + \sum_{D_k \notin \mathbf{D}'} 0 \cdot w_k = \sum_{D_i \in \mathbf{D}'} w_i.$$

V. Security Analysis

In this section, we will analyze the security of the proposed privacy-preserving range query scheme. We particularly focus on the privacy properties, i.e., the query range $[L_B, U_B]$ should be privacy-preserving, and the subset $\mathbf{D}'$ is also privacy-preserving in the proposed range query scheme.

The Query Range $[L_B, U_B]$ is Privacy-Preserving in the Proposed Scheme: As we know, in the proposed scheme, in order to achieve the communication efficiency, the query range $[L_B, U_B]$ has been represented as a matrix R of size $m \times m$, and the matrix R can be reconstructed by five binary vectors $(\bar{Y}_1, X'_1, X'_2, \bar{Y}_3, X'_3)$. Because BGN is semantically secure, and each element in vectors is encrypted with the BGN, without knowing the private key, no one can distinguish whether an encrypted element is encrypted from 0 or 1. As a result, the query range $[L_B, U_B]$ can be hidden, and the privacy-preserving requirement on the range query can be achieved in the proposed scheme.

The Subset $\mathbf{D}'$ Is Also Privacy-Preserving in the Proposed Scheme: In the proposed scheme, although each IoT device D_k does not know whether its sensed data w_k is in the query range, it can still hide w_k from the fog device, other IoT devices, and the query user. From the relationship $w_k = (i-1) \times m + j$, the IoT device D_k can determine i and j, and pick up $E(\bar{y}_{1j})$, $E(x'_{1i})$, $E(x'_{2i})$, $E(\bar{y}_{3j})$, and $E(x'_{3i})$ from five vectors $(\bar{Y}_1, X'_1, X'_2, \bar{Y}_3, X'_3)$. If D_k directly computes $c_k = e(E(\bar{y}_{1j}), E(x'_{1i})) \cdot e(E(x'_{2i}), g) \cdot e(E(\bar{y}_{3j}), E(x'_{3i}))$ and $s_k = c_k^{w_k}$, the fog device and other IoT devices can identify i, j by enumerating all possible combinations from $(\bar{Y}_1, X'_1, X'_2, \bar{Y}_3, X'_3)$, and D_k's data w_k will be disclosed. Luckily, because BGN has the property of self-blindness, when computing c_k and s_k, D_k also includes the random factors $e(g,h)^{r_{k_1}}$, $e(g,h)^{r_{k_2}}$. Thus,

neither the fog device nor other IoT devices can identify w_k. Since we consider there is no collusion between the fog device and the query user, the fog device will not forward individual c_k, s_k to the query user, instead, the fog device responds the aggregated results C and S to the query user. Therefore, each IoT device D_k's data w_k is also secure against the query user. As a result, no one, including the IoT device D_k itself, can know whether D_k is in the subset $\mathbf{D}'$ or not, and the subset $\mathbf{D}'$ is also privacy-preserving in the proposed scheme.

From the above analysis, we can see our proposed scheme is really a privacy-preserving range query scheme for Fog-enhanced IoT.

Remarks: We have noticed that several attacks against encrypted range queries over outsourced databases have been proposed in recent years [9]–[12]. In 2014, Islam *et al.* [9] used the access pattern disclosure to introduce an inference attack against encrypted range queries. In 2015, Naveed *et al.* [10] studied the property-preserving encryption schemes, such as deterministic (DTE) and order-preserving encryption (OPE), and presented a series of attacks that recover the plaintext from DTE- and OPE-encrypted database columns using only the encrypted column and publicly available auxiliary information. In 2016, Kellaris *et al.* [11] identified two basic sources of leakage, e.g., access pattern and communication volume, and developed generic reconstruction attacks on any system supporting range queries where either access pattern or communication volume is leaked. In 2017, Lacharité *et al.* [12] used the rang query leakage including access pattern, rank information, to propose improved reconstruction attacks on encrypted data. Different from the outsourced database scenarios discussed in [9]–[12], our proposed scheme is a privacy-preserving range query scheme for Fog-enhanced IoT scenario. More precisely, our proposed scheme is a special privacy-preserving aggregation scheme with a private range given in a query. Because our proposed scheme can preserve the privacy for both the query range $[L_B, U_B]$ and the subset $\mathbf{D}'$, no access pattern, rank information will be leaked. In addition, for any range query, the aggregated results (C, S) are of the same size, i.e., no communication volume information will be leaked. Therefore, our proposed scheme can defend against the latest attacks on secure range query [9]–[12].

VI. Performance Evaluation

In this section, we evaluate the performance of our proposed scheme in terms of computational costs and communication overheads. Specifically, we will compare our proposed scheme with the traditional BGN-based privacy-preserving range query scheme (denoted as PRQwo), which does not apply our range expression, decomposition, and composition technique, i.e., still using an array $A[1 \ldots n]$ for the range query as discussed in Section I. We implement both schemes with Java (JDK 1.8), JPBC library [13], where we use the type a1 pairings in JPBC which are constructed on the curve $y^2 = x^3 + x$ over the field $F_{\hat{q}}$ with $\hat{q} = 866510311086789487$ $561185306553193177862660915312774496609928941246404044834332308173042941398298 3$, and run our experiments on an Intel Core i7-7700 CPU@3.60 GHz Windows Platform with 16GB RAM. Note that, in our implementation, PRQwo only uses the homomorphic properties in $\mathbb{G}$, which does not require the pairing operation or the homomorphic properties in $\mathbb{G}_T$. The detailed parameter settings are shown in Table I. We run our experiment for 1000 times, and the average results are reported.

TABLE I
Parameter Settings

Parameter	Value
κ, p, q, $\mathcal{N}$	$\kappa = 512$, $\lvert p\rvert = \lvert q\rvert = \kappa = 512$, $\mathcal{N} = pq$
N	The number of IoT devices $N = 1000$
n	The upper bound of $[1, n]$, $n = \{10^2, 15^2, 20^2, 25^2, 30^2\}$
m	$m^2 = n$, $m = \{10, 15, 20, 25, 30\}$
Δ	The upper bound of $\mathbb{S} = \{0, 1, \cdots, \Delta\}$, $\Delta = 1000 \times 30^2$
$[L_B, U_B]$	The query bound $L_B \leq U_B$, randomly chosen from $[1, n]$
w_k	D_k's data randomly chosen from $[1, n]$

A. Computational Costs

In Fig. 4, we compare the average computational costs between the proposed scheme and PRQwo varying with n from 10^2 to 30^2. Fig. 4(a) depicts the case for query generation at query user. From the figure, we can see, with the increase of n, the computational cost of PRQwo linearly increases with n, while the cost increase in our proposed scheme follows the square-root law of n and is very low. Therefore, only sending five vectors $(\bar{Y}_1, X'_1, X'_2, \bar{Y}_3, X'_3)$ in our proposed scheme is much more efficient in terms of range query generation. Fig. 4(b) shows the case for query response at IoT device. From the figure, we can see the computational costs in both schemes are efficient and irrelevant with n. However, PRQwo is more efficient than our proposed scheme. The reason is that our proposed scheme needs to pick up five elements and run time-consuming pairing operations, while PRQwo just needs to pick up one element and does not include the pairing operation. Fig. 4(c) plots the case for the response aggregation at fog device. From the figure, we can observe the computational costs in both schemes are independent of n, because they are only related to the number of IoT devices N. In addition, since the multiplication is executed over $\mathbb{G}_T$, the proposed scheme is more efficient than PRQwo. Fig. 4(d) indicates the case for the response recovery at query user. From the figure, we can see both schemes are efficient and irrelevant to n. At the same time, since the decryption is executed over $\mathbb{G}_T$, the proposed scheme is also more efficient than PRQwo.

B. Communication Overhead

The novelty of this paper is that we make use of the range expression, decomposition, and composition technique to achieve the communication efficiency. In Fig. 5, we compare the communication overhead (from the query user to the fog device) between our proposed scheme and PRQwo varying from n. From the figure, we can see the communication overhead in PRQwo increases linearly with n, while the overhead in our proposed scheme is much more efficient, which can achieve $O(\sqrt{n})$ communication efficiency.

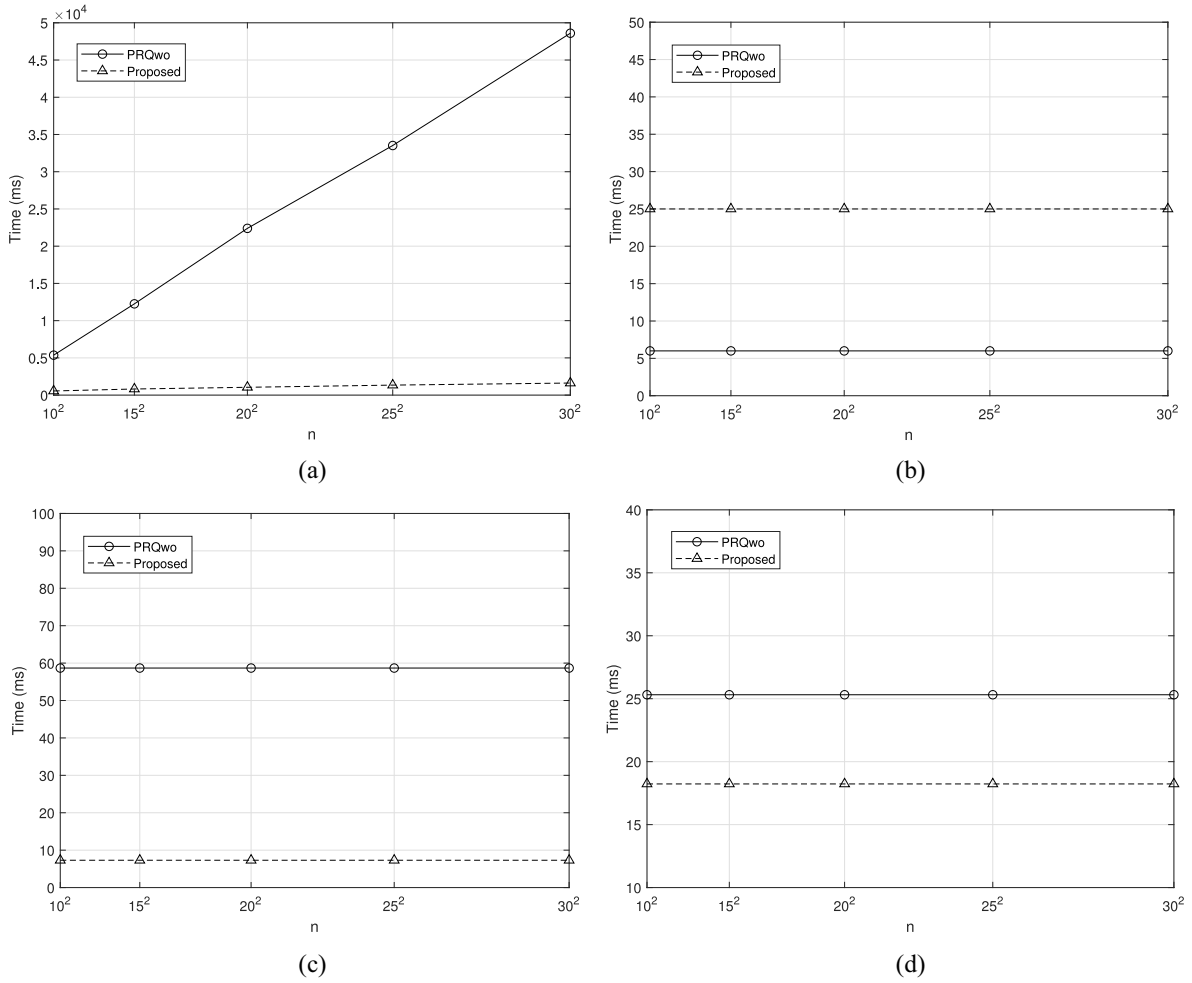

Fig. 4. Computational cost comparisons between the proposed scheme and PRQwo varying with n. (a) Query generation at query user. (b) Query response at IoT device. (c) Response aggregation at fog device. (d) Response recovery at query user.

From the above evaluation, our proposed scheme indeed achieves the communication-efficient privacy-preserving range query in Fog-enhanced IoT.

VII. Related Work

In this section, we briefly review some recently proposed privacy-preserving range query schemes in outsourced cloud computing [14]–[17], tiered wireless sensor networks [18], [19], and participatory sensing [20] scenarios.

Samanthula and Jiang [14] proposed efficient privacy-preserving range queries over encrypted data in cloud computing. As secure comparison of encrypted integers are critical for evaluating the range queries, an efficient method for converting an encrypted integer z into encryptions of the individual bits of z is proposed for supporting secure comparison in [14]. In [15], in order to achieve secure and efficient range queries on outsourced data, Wang and Ravishankar used the asymmetric scalar-product preserving encryption to build $\hat{R}$ tree for encrypted halfspace range query in $\mathbb{R}^d$. Li *et al.* [16] designed PBtree data structure and associated algorithms for PRtree construction, searching, optimization, and update to support real-time range queries with strong privacy protection in cloud. Shen *et al.* [17] studied the problem

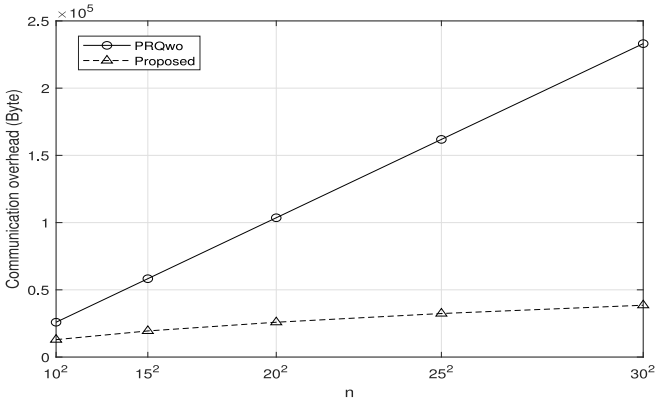

Fig. 5. Communication overhead (from the query user to the fog device) comparisons between the proposed scheme and PRQwo varying with n.

of multidimensional private range queries over outsourced cloud data, their proposed scheme can achieve personalized search with flexible privacy and high efficiency according to owner-specified privacy-cost tradeoff. In tiered wireless sensor network scenarios, Zhang *et al.* [18] used the bloom filter technique to propose an efficient and secure range query protocol, which can not only preserve the privacy of data and queries

but also verify the integrity of results. Zeng *et al.* [19] applied the distance comparison rule to propose a privacy-preserving, energy-efficient, and multidimensional range query protocol, which not only achieves data privacy but also considers collusion attacks, probability attacks, and differential attacks. In participatory sensing scenario, Zeng *et al.* [20] also used the distance comparison rule to propose a high energy-efficient and privacy-preserving range query framework, which can balance energy-efficient, privacy-preserving, and data reliability in participatory sensing.

Different from the above works, our proposed scheme focuses on privacy-preserving range query in Fog-enhanced IoT, which does not rely on secure comparison of encrypted integers, and can protect not only the query range $[L_B, U_B]$ but also the subset $\mathbf{D}'$. In addition, compared with the straightforward PRQwo scheme mentioned in Section I, our proposed scheme can achieve $O(\sqrt{n})$ communication efficiency.

VIII. Conclusion

In this paper, we have proposed a new communication-efficient privacy-preserving range query scheme in Fog-enhanced IoT. The proposed scheme is characterized by employing BGN homomorphic encryption and a range query expression, decomposition, and composition technique to not only preserve the privacy for both the range query and individual IoT device's data but also achieve the $O(\sqrt{n})$ communication efficiency. Detailed security analyses show the proposed scheme is really privacy-preserving under our defined security model. In addition, extensive performance experiments are conducted, and the results indicate it is really efficient in terms of low range query generation cost and low communication overhead. In our future work, we will explore more general privacy-preserving range query functions, e.g., multirange query, in Fog-enhanced IoT.

References

[1] *Gartner Says 8.4 Billion Connected 'Things' Will Be in Use in 2017, Up 31 Percent From 2016*, Gartner, Stamford, CT, USA, 2017. [Online]. Available: https://www.gartner.com/newsroom/id/3598917

[2] 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.

[3] K.-K. R. Choo, R. Lu, L. Chen, and X. Yi, "A foggy research future: Advances and future opportunities in fog computing research," *Future Gener. Comput. Syst.*, vol. 78, pp. 677–679, Jan. 2018.

[4] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, "A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT," *IEEE Access*, vol. 5, pp. 3302–3312, 2017.

[5] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts," in *Proc. TCC*, Cambridge, MA, USA, Feb. 2005, pp. 325–341.

[6] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Adv. Cryptol. EUROCRYPT Int. Conf. Theory Appl. Cryptograph. Tech. Prague Czech Republic*, May 1999, pp. 223–238.

[7] T.-H. H. Chan, E. Shi, and D. Song, "Privacy-preserving stream aggregation with fault tolerance," in *Proc. 16th Int. Conf. Financ. Cryptography Data Security (FC)*, Kralendijk, Bonaire, Feb./Mar. 2012, pp. 200–214.

[8] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1997.

[9] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Inference attack against encrypted range queries on outsourced databases," in *Proc. 4th ACM Conf. Data Appl. Security Privacy (CODASPY)*, San Antonio, TX, USA, Mar. 2014, pp. 235–246.

[10] M. Naveed, S. Kamara, and C. V. Wright, "Inference attacks on property-preserving encrypted databases," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Security*, Denver, CO, USA, Oct. 2015, pp. 644–655.

[11] G. Kellaris, G. Kollios, K. Nissim, and A. O'Neill, "Generic attacks on secure outsourced databases," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Vienna, Austria, Oct. 2016, pp. 1329–1340.

[12] M.-S. Lacharité, B. Minaud, and K. G. Paterson, "Improved reconstruction attacks on encrypted data using range query leakage," IACR Cryptol. ePrint Archive, Rep. 2017/701, 2017.

[13] A. D. Caro. *The Java Pairing Based Cryptography Library (JPBC)*. Accessed: Jun. 4, 2018. [Online]. Available: http://libeccio.di.unisa.it/projects/jpbc/

[14] B. K. Samanthula and W. Jiang, "Efficient privacy-preserving range queries over encrypted data in cloud computing," in *Proc. IEEE 6th Int. Conf. Cloud Comput.*, Santa Clara, CA, USA, Jun./Jul. 2013, pp. 51–58.

[15] P. Wang and C. V. Ravishankar, "Secure and efficient range queries on outsourced databases using Rp-trees," in *Proc. 29th IEEE Int. Conf. Data Eng. (ICDE)*, Brisbane QLD, Australia, Apr. 2013, pp. 314–325.

[16] R. Li, A. X. Liu, A. L. Wang, and B. Bruhadeshwar, "Fast and scalable range query processing with strong privacy protection for cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2305–2318, Aug. 2016.

[17] Y. Shen, L. Huang, and W. Yang, "Achieving personalized and privacy-preserving range queries over outsourced cloud data," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.

[18] X. Zhang *et al.*, "Achieving efficient and secure range query in two-tiered wireless sensor networks," in *Proc. IEEE 22nd Int. Symp. Qual. Service (IWQoS)*, Hong Kong, May 2014, pp. 380–388.

[19] J. Zeng *et al.*, "Privacy-preserving and multi-dimensional range query in two-tiered wireless sensor networks," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, Singapore, Dec. 2017, pp. 1–7.

[20] J. Zeng *et al.*, "Energy-efficient and privacy-preserving range query in participatory sensing," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Tianjin, China, Aug. 2016, pp. 876–883.

Rongxing Lu (S'09–M'11–SM'15) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012.

He was an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, from 2013 to 2016. He has been an Assistant Professor with the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Fredericton, NB, Canada, since 2016. He was a Post-Doctoral Fellow with the University of Waterloo, from 2012 to 2013.

Dr. Lu was a recipient of the Governor General's Gold Medal for his Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, the 8th IEEE Communications Society (ComSoc) Asia–Pacific Outstanding Young Researcher Award in 2013, and the 2016 to 2017 Excellence in Teaching Award from FCS, UNB. He is currently serves as the Vice-Chair (Publication) of IEEE ComSoc CIS-TC.