Using Reduced Paths to Achieve Efficient Privacy-Preserving Range Query in Fog-Based IoT

Hassan Mahdikhani, *Graduate Student Member, IEEE*, Rongxing Lu, *Senior Member, IEEE*, Jun Shao, and Ali Ghorbani, *Senior Member, IEEE*

***Abstract*—The fog computing architectural model has recently seen advances with respect to bandwidth and latency issues. However, since fog devices are deployed at the network edge and are not fully trustable, there are still security and privacy challenges. In this article, aiming at improving both communication efficiency and privacy protection, we propose a new efficient and privacy-preserving range query scheme in fog-based Internet of Things (IoT). We, first, introduce a new decomposition technique to efficiently interpret a given range query $[L, U]$, where $0 \leq L \leq U \leq n-1$, as a form of inverted reduced path strings. Then, the symmetric homomorphic encryption (SHE) scheme is employed to encrypt the reduced paths and hand them over securely through a fog node to the IoT devices. This technique enables a query user to launch a privacy-preserving continuous or noncontinuous range query and receive a homomorphically aggregated encrypted response with an improved $O(\log^2 n)$ communication efficiency. The detailed security analysis shows that our proposed scheme is privacy preserving. In addition, extensive performance evaluations are also conducted, and the results demonstrate that our proposed scheme is by far more efficient than those previously reported schemes in terms of computational overhead and communication complexity.**

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

I. Introduction

RECENT advancements in the Internet of Things (IoT) and wireless communication technologies have enabled us to enjoy various smart things around our daily lives, including smart vehicles [1]–[3], smart homes [4]–[6], smart grids [7]–[10], and smart healthcare [11]. All of these are highly dependent upon the massive data generated by IoT. However, due to the limited built-in storage and computing capability of IoT devices, the data usually need to be transferred from IoT devices to cloud serves for storage and processing, which will inevitably encounter bandwidth problems and incur response latency [12]. In order to address these bandwidth and latency issues, the concept of fog computing was proposed by Cisco in 2012 [13]. Essentially, the idea is to deploy fog devices equipped with certain storage and computing capabilities at the network edge, which can directly solve problems or preprocess parts of a problem close to the IoT domain. Thus, it can provide low latency responses and improve communication efficiency by only forwarding preprocessed data to cloud server. Over the past years, IoT integrated with fog computing has attracted considerable attention. However, since fog devices are deployed at the network edge and might not be fully trusted, fog-based computing still faces some security and privacy issues [14], [15]. In this article, we aim to insure privacy while also being efficient when performing privacy-preserving range aggregate queries in fog-based IoT scenarios.

To clearly illustrate the research problem in fog-based IoT, let us consider the following scenario in a smart grid [8], [9]. For the better electricity planning, the manager (i.e., a query user) of a smart grid may query the number of houses whose individual electricity consumption is within a certain range in a certain area for planning purposes. If neither security nor privacy are of concerns, this query can be easily completed. The query user sends the range to the fog devices and the fog devices can directly return the query responses through comparing the range with the values from the IoT devices (i.e., smart meters). However, when the privacy is taken into consideration, the values from the meters may reveal the resident's privacy, such as whether he/she lived in the house recently. Moreover, the query and the corresponding result may also reveal the query user's privacy. For example, his/her selling strategy could be deduced from the query and result. For reasons such as these, several solutions have been proposed [16], [17]. However, all of them suffer from one or more of the following problems.

1) *Privacy Level:* The fog devices in many solutions [18]–[20] can deduce the query result from the communications between the query user and IoT devices.
2) *High Communication Overhead:* The communication complexity in a straightforward solution [16] is $O(n)$, where n is the biggest value that the smart meter can return in a certain period of time. The current best result is $O(\log^3 n)$ communication efficiency, which seems still high for large n [17].

Manuscript received August 23, 2020; accepted September 28, 2020. Date of publication October 7, 2020; date of current version March 5, 2021. This work was supported in part by NSF of Zhejiang Province under Grant LZ18F020003, and in part by NSFC under Grant U1709217. The work of Rongxing Lu was supported by the National Science and Engineering Research Council of Canada (NSERC) under Grant Rgpin 04009. The work of Ali Ghorbani was supported by the NSERC through the Discovery Grant and Canada Research Chair. *(Corresponding author: Rongxing Lu.)*

Hassan Mahdikhani, Rongxing Lu, and Ali Ghorbani are with the Canadian Institute for Cybersecurity, Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: hmahdikh@unb.ca; rlu1@unb.ca; ghorbani@unb.ca).

Jun Shao is with the School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou 310018, China (e-mail: chn.junshao@gmail.com).

Digital Object Identifier 10.1109/JIOT.2020.3029472

3) *Limited Query Types:* Most of the current solutions cannot support noncontinuous range queries directly, and the only work around is to run the original solution repeatedly. However, the resulting solution would reveal some private information to the fog device, e.g., how many ranges are in the query.

Aiming to solve the above issues, in this article, we employ a reduced path technique to implement a new efficient and privacy-preserving range query scheme for fog-based IoT. Specifically, the contributions of this article are threefold.

1) We present a new privacy-preserving range query solution for the fog-based IoT. The notable property of our proposal is that the communication complexity is only $O(\log^2 n)$. To the best of our knowledge, it is the best result for privacy-preserving range queries in the fog-based IoT setting.
2) Our proposal is the first to support continuous and noncontinuous queries and aggregating queries at the same time. The decomposition technique addresses query format of recent studies [16], [17]. Also, [17] suffers from a limitation in the query form in which the lower and upper bound values must be of the form of two to the power of integer value, i.e., $[L, U] = [2^a, 2^{a'}]$. The critical component, which may be of independent interest, is the use of encrypted inverted reduced paths. Range queries are decoded as reduced paths in a perfect binary tree (PBTree), where all the nodes are encrypted with the symmetric homomorphic encryption (SHE) scheme proposed in our previous study [17].
3) The detailed security analysis shows that our solution is indeed a privacy-preserving scheme. Furthermore, extensive experiments also demonstrate that our proposal outperforms state-of-the-art solutions in terms of low communication complexity [16], [17].

The remainder of this article is structured as follows. We first formalize our system model, security model, and design goal in Section II. Then, we recall our preliminaries in Section III. After that, we present our proposed scheme in Section IV, followed by security analysis and performance evaluation in Sections V and VI, respectively. Some related works are discussed in Section VII, and finally, conclusions are drawn in Section VIII.

II. Models and Design Goal

In this section, we formalize our system model, security model, and set out the design goal for our communication-efficient privacy-preserving range query. For a clear description, some used symbols are first listed in Table I.

A. System Model

In our system model for privacy-preserving range queries in fog-based IoT, as shown in Fig. 1, there are three kinds of entities, namely, a query user at the user layer, a fog node at the fog layer, and a set of IoT end devices $\mathbf{I} = \{I_1, I_2, \ldots, I_N\}$ at the end-device layer. The detailed descriptions of these entities are as follows.

Query User: In our system model, a query user formulates and submits secure range queries to the IoT end devices via the fog node, and also obtains the corresponding encrypted responses via the fog node. Particularly, as indicated in Fig. 1, the range count query, i.e., COUNT($\mathbf{I}'$), can be submitted to retrieve the number of IoT end devices in subset $\mathbf{I}' \subseteq \mathbf{I}$. Formally, we consider $\mathbf{I}'$ as a subset of $\mathbf{I} = \{I_1, I_2, \ldots, I_N\}$, where the prepared data w_i at each IoT end device $I_i \in \mathbf{I}'$ is within the range $[L, U]$, i.e.,

$$\mathbf{I}' = \{I_i | I_i \in \mathbf{I} \ \wedge w_i \in [L, U]; 0 \le L \le U \le n-1\}. \quad (1)$$

Upon receiving the above count query, the fog node returns the response as follows:

$$\text{COUNT}(\mathbf{I}') = \text{the size}|\mathbf{I}'|. \quad (2)$$

Note that the range $[L, U]$ in our proposed system model may be continuous or noncontinuous, i.e.,

$$[L, U] = [L_1, U_1] \cup [L_2, U_2] \cup \cdots \cup [L_s, U_s] = \bigcup_{\iota=1}^{s} [L_\iota, U_\iota]. \quad (3)$$

For example, our proposed scheme can handle simple continuous range queries, e.g., $[L, U] = [5, 12]$, and complex noncontinuous ones, e.g., $[L, U] = [0, 7] \cup [16, 19] \cup [24, 24] \cup [28, 29]$, as depicted in Figs. 2 and 3, respectively.

Fog Node: A fog node is deployed at the network edge, i.e., the fog layer, which processes data prepared by IoT end devices and relays the aggregated results to the query user.

TABLE I
List of Symbols Used in Proposed Scheme

Symbol	Description
N	The number of the IoT devices.
$\mathbf{I}, I_i$	The i-th IoT device in set $\mathbf{I} = \{I_1, I_2, \cdots, I_N\}$, $1 \le i \le N$.
n	The maximum possible value that can be sensed by IoT devices.
w_i	Sensed and prepared data in I_i ranging from 0 to $n-1$; $0 \le w_i \le n-1$.
L, U	Lower and upper bound in a range query, $0 \le L \le U \le n-1$.
$\mathbf{I}'$	The subset of $\mathbf{I}$ where I_i's data $w_i \in [L, U]$.
PP	Public parameter in SHE scheme, i.e., $PP = (k_0, k_1, k_2, \mathcal{N})$.
SK	Secure secret key in SHE scheme; $SK = (p, q, \mathcal{L})$.
$E(), D()$	Encryption and Decryption functions.
T	The corresponding perfect binary tree with the whole range $[0, n-1]$.
S	Paths from root to leaves in T where leaf node $t \in [L, U]$.
R	Reduced path strings, extracted from S.
V	Inverted path strings, by inverting 0s to 1s and vice versa in R.
Q	Ciphertext equivalent of V, by separately encrypting 0s and 1s in V.
F	Final encrypted range query including real paths, dummy paths, and filtering column.

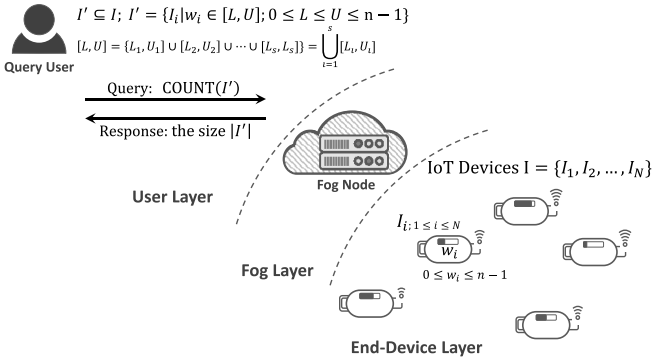


Fig. 1. System model under consideration.

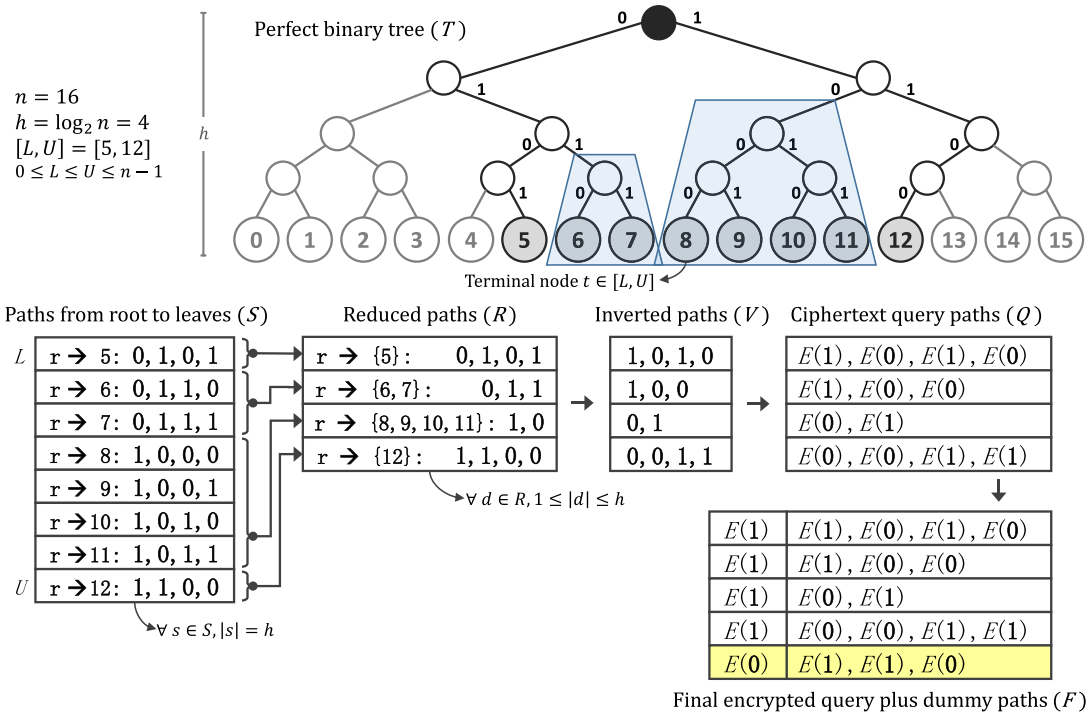


Fig. 2. Example of converting a simple continuous range query into encrypted reduced paths.

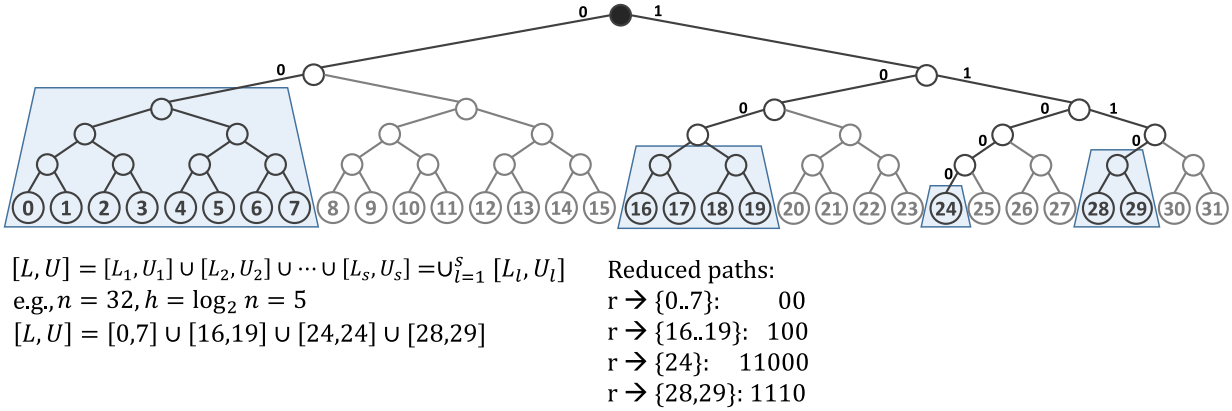


Fig. 3. Noncontinuous range query consists of two or more individual ranges and its corresponding reduced paths.

In this model, an encrypted query from the query user is forwarded to IoT end devices via the fog node. The IoT end devices, then, encrypt their responses and report them to the fog node for aggregation. Finally, the fog node returns the aggregated results to the query user.

IoT End Devices $\mathbf{I} = \{I_1, I_2, \ldots, I_N\}$: A large population of IoT end devices ($\mathbf{I}$) of size N are spread across the end-device layer and related to the specific IoT ecosystem. After collecting the raw data w_i from sensing component, each end-device I_i can provide data processing functionality before delivering the data to the fog node. More precisely, different processing tasks, such as validating, converting, packaging, and even encrypting will enable IoT end device I_i to successfully transmit highly accurate sensitive data. For example, suppose a sensed high-precision floating-point value $w_i = 0.195874$ is to be send. It can be modified into an arbitrary precision integer by applying simple scaling, truncating, or rounding functions, e.g., truncate$(w_i = 0.195874 * 10000 = 1958.74) = 1958$. Values are then encrypted and sent to the fog node. For simplicity but without losing generality, we assume that the prepared data w_i are converted into integer values and lie within the range of $[0, n - 1]$, where n is a power of two. Note that we can easily extend the range so that the condition $n = 2^h$ holds. Continuing with our example, we can initialize $n = 2048 = 2^{11}$ to appropriately cover all possible return values w_i in the range $0 \le w_i \le 2048$.

B. Security Model

In our security model, all entities are assumed to be honest-but-curious participants, i.e., they faithfully follow the query processing protocols but might try and extract any additional query information during the data preparation and throughout the query processing steps. For example, the fog node may be curious about each IoT device's data w_i and the user's query

range $[L, U]$; each IoT device may be curious about other IoT devices' data and the query range $[L, U]$; and the query user may be curious about each IoT device data w_i, other than query response value Count($\mathbf{I}'$). We also assume there is no collusion between any two entities in our model.

Note that an external adversary may launch active attacks on data integrity and source authentication, but as this work focuses on communication efficiency and privacy those attacks are beyond the scope of this article and left for future work.

C. Design Goals

Given the above system and security models, our design goals are to present a privacy-preserving and communication-efficient range query scheme for fog-based IoT.

Proposed Scheme Should Be Privacy Preserving: Both the lower and upper bound values in the query $[L, U]$ should be private, i.e., no one, except the user, can retrieve $[L, U]$ to trace the query user's interests. In addition, the data collected by IoT devices are also sensitive and should be private, i.e., those data must only be accessible by their owners. Neither the fog node nor the query user is permitted to retrieve/recover the plain value w_i of each IoT device I_i.

Proposed Scheme Should Be Efficient: Energy consumption is the critical bottleneck in IoT ecosystems and, at the same time, reaching the above privacy goal will introduce additional communication overhead. Therefore, in the proposed scheme, we aim to improve the communication efficiency by significantly reducing the number of encrypted blocks and achieving $O(\log^2 n)$ communication efficiency, much better than previously reported ones [16], [17].

III. Preliminaries

In order to accomplish the above communication-efficient privacy-preserving range query scheme in fog-based IoT, we need to adopt homomorphic encryption techniques, which support both homomorphic addition and multiplication. Since most of the existing homomorphic encryption schemes are in the public-key settings and computationally inefficient, we briefly revisit the SHE scheme [17] to leverage it as a fundamental building block for our privacy-preserving range query. It should be noted that the SHE scheme differs from the order revealing encryption (ORE) technique [21]–[23] that enables searchable encryption instead of homomorphic aggregation. We also review the XOR operation and introduce our homomorphic XOR operator to check equality/inequality of encrypted bit values as it is the core operation of our proposed scheme.

A. Description of SHE Scheme

The SHE scheme, which is secure under the known-plaintext attack, has three algorithms, namely, *key generation*, *encryption*, and *decryption* [17], which is described as follows.

1) *Key Generation:* Given the security parameters (k_0, k_1, k_2) satisfying $k_1 \ll k_2 < (k_0/2)$, generate the secret key $SK = (p, q, \mathcal{L})$, where p and q are two large prime numbers with $|p| = |q| = k_0$ and $\mathcal{L}$ is a random number with the bit length $|\mathcal{L}| = k_2$. Compute $\mathcal{N} = pq$ and set the public parameter $PP = (k_0, k_1, k_2, \mathcal{N})$. At the same time, set the message space $\mathcal{M}$ as $\{0, 1\}^{k_1}$.
2) *Encryption:* A message $m \in \mathcal{M}$ can be encrypted with the secret key $SK = (p, q, \mathcal{L})$ as
$$c = E(m) = (r\mathcal{L} + m)\big(1 + r'p\big) \bmod \mathcal{N} \tag{4}$$
where $r \in \{0, 1\}^{k_2}$ and $r' \in \{0, 1\}^{k_0}$ are two random numbers.
3) *Decryption:* A ciphertext $c = E(m)$ can be decrypted with the secret key $SK = (p, q, \mathcal{L})$ as
$$D(c) : m = (c \bmod p) \bmod \mathcal{L}. \tag{5}$$
The correctness of the decryption is as follows:
$$\begin{aligned} D(c) &= (c \bmod p) \bmod \mathcal{L} \\ &= \big(\big((r\mathcal{L} + m)(1 + r'p) \bmod \mathcal{N}\big) \bmod p\big) \bmod \mathcal{L} \\ &= (r\mathcal{L} + m) \bmod \mathcal{L} \quad (\because 2k_2 < k_0) \\ &= m \quad (\because k_1 \ll k_2). \end{aligned}$$

Given the public parameter PP, SHE enjoys the following homomorphic properties.

1) *Homomorphic Addition-I:* Given two ciphertexts $c_1 = E(m_1) = (r_1\mathcal{L}+m_1)(1+r'_1p) \bmod \mathcal{N}$ and $c_2 = E(m_2) = (r_2\mathcal{L} + m_2)(1 + r'_2p) \bmod \mathcal{N}$, we have $c_1 + c_2 \rightarrow E(m_1 + m_2)$.
2) *Homomorphic Multiplication-I:* Given two ciphertexts c_1 and c_2, we have $c_1 \cdot c_2 \rightarrow E(m_1 \cdot m_2)$.
3) *Homomorphic Addition-II:* Given a ciphertext $c_1 = E(m_1) = (r_1\mathcal{L} + m_1)(1 + r'_1p) \bmod \mathcal{N}$, and a plaintext m_2, we have $c_1 + m_2 \rightarrow E(m_1 + m_2)$.
4) *Homomorphic Multiplication-II:* Given a ciphertext c_1 and a plaintext m_2, we have $c_1 \cdot m_2 \rightarrow E(m_1 \cdot m_2)$.

Based on the above homomorphic properties of SHE, we design our efficient privacy-preserving range query scheme in the next section.

B. XOR Gate

One of the most useful digital logic gates to compare binary values is the XOR gate ($\oplus$), i.e., $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, and $1 \oplus 1 = 0$. In our proposed scheme, we need to homomorphically compare pairs of encrypted zeros and ones. Therefore, for the given SHE ciphertext values $E(X)$ and $E(Y)$, the following homomorphic expression must be evaluated to securely obtain $E(X \oplus Y)$:
$$E(X \oplus Y) = E(X) + E(Y) - 2E(X)E(Y). \tag{6}$$

IV. Our Proposed Scheme

Our $O(\log^2 n)$ communication-efficient privacy-preserving range query scheme for fog-based IoT has roots in both novel path reduction process and SHE homomorphic encryption. We first describe our path reduction process, which is executed at the query user side to generate the encrypted reduced paths from a given range query $[L, U]$ over the range $[0, n - 1]$. Then, we describe our privacy-preserving range query scheme in detail: 1) initializing the SHE cryptosystem (key generation phase); 2) generating encrypted range query at user

side; 3) responding to the encrypted query at IoT device; 4) aggregating the encrypted responses in the fog node; and 5) decrypting the aggregated response to get the final result at user side.

A. Range Queries as Reduced Paths

Our $O(\log^2 n)$ communication-efficient algorithm formulates the range query $[L, U]$, where $0 \le L \le U \le n-1$ and n is a power of two, as a reduced path structure in which the length of each entry lies between 1 and $h = \log_2 n$. This process consists of the following steps.

1) Construct Perfect Binary Tree: Given a range query $[L, U]$ over $[0, n-1]$, the PBTree T is constructed in which the leaf nodes are sequentially labeled from left to right with numbers from zero to $n-1$. Moreover, the left edges of T are labeled by zeros and the right edges by ones. For example, Fig. 2 depicts the sample PBTree T for $n = 16$ and simple continuous range $[L, U] = [5, 12]$.

Notice that this scheme, unlike [16] and [17], supports not only simple continuous range queries $[L, U]$ but also noncontinuous range queries that consist of two or more nonadjacent range queries, i.e., $[L, U] = \cup_{\iota=1}^{s}[L_\iota, U_\iota] = [L_1, U_1] \cup [L_2, U_2] \cup \cdots \cup [L_s, U_s]$. For example, a noncontinuous range query and its corresponding reduced paths are depicted in Fig. 3.

Definition 1 (PBTree): A PBTree T with n leaf nodes has $n-1$ remaining nonterminal nodes, including a root node and $n-2$ internal nodes. It can be declared as below in which all nonterminal nodes have both left and right nonempty subtrees, and all leaves have the same height $h = \log_2 n$, where $n \ge 2$ is a power of two. In addition, edges leading to the left and right subtrees are, respectively, labeled with zeros and ones

```
public class PBTree {
    Node root;
    static class Node {
        int data;
        Node left, right;
        Node(int data) {
            this.data = data;
            this.left = null;
            this.right = null;
        }
}
```

2) Range Query Encoding: In this step, terminal node t in range query $[L, U]$ is encoded into a binary string s that represents the path from the root to the terminal node t. To this end, Algorithm 1 recursively traverses from root to all leaf node t in $[L, U] = \cup_{\iota=1}^{s}[L_\iota, U_\iota]$ and uses the visited array to keep track of the discovered vertices. Therefore, at the end of this traversal process, there exists the entry $s \in S$ that indicates the corresponding path from root to leaf nodes t in $[L, U]$. Moreover, the size of S is $U - L + 1$, i.e., $|S| = U - L + 1$. Also, the length of each entry in S equals the height of PBT T, i.e., $|s \in S| = \log_2 n = h$. For example, in Fig. 2, the path from the root to terminal node $t = 11 \in [5, 12]$ can be expressed as "r, l, r, r," equivalent to $s =$ "1011" by simply replacing "r(*right*)"s with "1"s and "l(*left*)" with "0". Also, note that the length of each entry $s \in S$ is $|s| = h = \log_2 16 = 4$.

3) Reducing Paths: To achieve $O(\log^2 n)$ communication efficiency, our path reduction algorithm combines the listed paths in S and forms the reduced paths R. Fig. 2 illustrates an example of our path reduction technique output and Algorithm 2 describes in detail how to merge the distinct paths to form R from S. In each iteration of the loop in our reduction algorithm, two leaf nodes or sibling subtrees with corresponding path strings $\mathcal{P}$ and $\mathcal{P}'$ are reduced if: 1) they are both of the same length (ℓ); 2) their prefix substrings of length $\ell - 1$ are equal; and 3) they differ only in their last entry. This process continues until reaching the maximum possible pruning and stops when there are no more adjacent path strings to be reduced ($\mathcal{P}$ and $\mathcal{P}'$ are zero-based indices)

$$\left(|\mathcal{P}| = |\mathcal{P}'| = \ell\right) \wedge \left(\mathcal{P}_{[0..\ell-2]} = \mathcal{P}'_{[0..\ell-2]}\right) \wedge \left(\mathcal{P}_{[\ell-1]} = \mathcal{P}'_{[\ell-1]}\right). \tag{7}$$

For example, the path from root to two different leaf nodes five ("0101") and six ("0110") are not reducible due to their different prefix substrings of length $\ell - 1 = 3$. However, as shown in the figure, the two subtree sibling nodes six ("0110") and seven ("0111") are reducible and they will be merged into single path string "011" as they follow our reduction rule.

4) Inverting Reduced Paths: Since the XOR operation will eventually be applied to check whether an IoT device's prepared data are in the range query or not, the reduced paths R are flipped to get the inverted paths V. As shown in Fig. 2, inverted paths V are simply obtained by flipping zeros to ones and ones to zeros in each string entry of R.

5) Encrypting Inverted Paths: Inverted paths V, in turn, are encrypted to form the ciphertext query paths Q. To do so, we simply encrypt zeros and ones by using the SHE scheme.

Algorithm 1 Range Query to Traversal Path Strings

Input: PBTree T, $[L, U] = \cup_{\iota=1}^{s}[L_\iota, U_\iota]$; $0 \le L \le U \le n-1$
Output: ArrayList S ▷ Paths from root to leaf terminal nodes that belong to the range query

```
PathToLeaf ← Ø                          ▷ ArrayList<Character> PathToLeaf
S ← Ø                                   ▷ ArrayList<String> S
for each t ∈ [L, U] do
    T.GETPATH(T.root, PathToLeaf, t, ' ')
    S.ADD(PathToLeaf)
    PathToLeaf.CLEAR()
end for
return S
    ▷ RT=root, PTNode=PathToNode, DNode=DestinationNode, VEdge=VisitedEdge
procedure GETPATH(RT, PTNode, DNode, VEdge)
    if RT = null then
        return false
    end if
    PTNode.ADD(VEdge)                   ▷ Accumulate the visited edges
    if RT.data = DNode then
        return true
    end if
    if GETPATH(RT.left, PTNode, DNode, '0') ∨
    GETPATH(RT.right, PTNode, DNode, '1') then   ▷ 0 for traversing left subtrees and 1 for right ones
        return true
    end if
    PTNode.REMOVE(PTNode.SIZE() − 1)
    return false
end procedure
```

Algorithm 2 Reducing Paths

Input: ArrayList S, $|s \in S| = h$ ▷ Paths from root to leaf nodes
Output: ArrayList R, $1 \leq |d \in R| \leq h$ ▷ Reduced paths

```
1: ArrayList RPath[2]                                   ▷ RPath[0] and RPath[1]
2: RPath[0] ← S
3: Src ← 0, Dst ← 1                  ▷ Determine the source and destination RPath
4: flag ← false                                         ▷ Terminate the loop execution
5: repeat
6:     flag ← false
7:     for each d_i ∈ RPath[Src] do                     ▷ 0 ≤ i < RPath[Src].SIZE()
8:         if i < RPath[Src].SIZE() − 1 then
9:             strRes ← REDUCE(RPath[Src].GET(i), RPath[Src].GET(i + 1))
10:            if strRes = " " then
11:                RPath[Dst].ADD(RPath[Src].GET(i))
12:            else
13:                RPath[Dst].ADD(strRes)
14:                flag ← true
15:                i++
16:            end if
17:        else
18:            RPath[Dst].ADD(RPath[Src].GET(RPath[Src].SIZE() − 1))
19:        end if
20:    end for
21:    if flag = true then                              ▷ Plan for the next iteration
22:        RPath[Src].CLEAR()
23:        Src = (Src + 1) mod 2
24:        Dst = (Dst + 1) mod 2
25:    end if
26: until flag = true
27: R ← RPath[Dst]
28: return R

29: procedure REDUCE(x, y)
30:    ℓ_x ← x.LENGTH()
31:    ℓ_y ← y.LENGTH()
32:    if ((ℓ_x = ℓ_y) ∧ (x_[0..ℓ_x−2] = y_[0..ℓ_y−2]) ∧ (x_[ℓ_x−1] ≠ y_[ℓ_y−1])) then
33:        return x_[0..ℓ_x−2]
34:    end if
35:    return " "
36: end procedure
```

6) Adding Dummy Paths: Let us consider what happens when the query user examines the range query: $[L, U] = [0, (n/2) - 1]$ (namely, the entire left subtree). The reduced path, in this case, is "0" and consequently the encrypted inverted path contains just one entry, including one encrypted value, i.e., "$E(1)$." However, transmitting it to the fog node would lead to a high probability of inferring the query. For instance, in our above example, by simply submitting ciphertext "$E(1)$," the fog node can infer that the query user has examined one of the following queries: the left subtree ($[L, U] = [0, (n/2) - 1]$) or right subtree ($[L, U] = [(n/2), n - 1]$). To overcome this problem, dummy paths (shaded yellow row in Fig. 2) are injected to prevent the curious fog node from inferring or guessing the exact range query. They are appended to encrypted query paths Q to output final encrypted query F. The theoretical analysis on dummy paths' length and guessing probability is detailed later in Section V. Finally, an additional encrypted column is added to each path entry to distinguish the dummy paths from the real ones. $E(1)$ will be assigned to valid paths and $E(0)$ to dummy paths. Therefore, since the dummy paths are multiplied by $E(0)$, they will be implicitly discarded during the aggregation phase in the fog node.

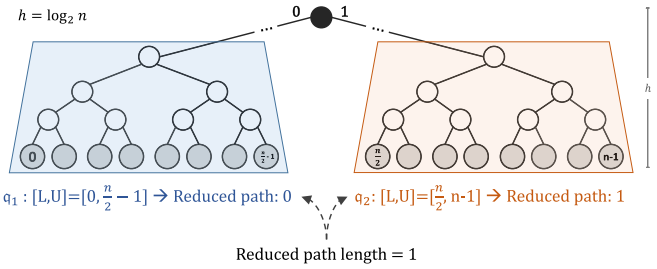


Fig. 4. Lower bound communication complexity (minimum reduced path length) with q_1 ad q_2 that have, respectively, queried left and right subtrees.

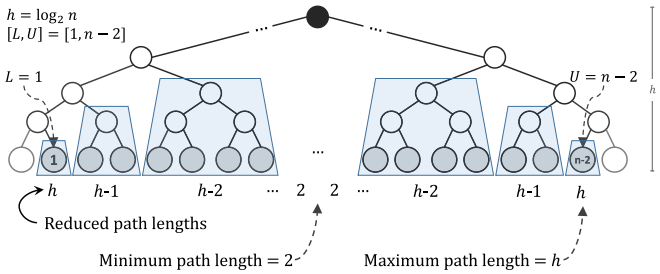


Fig. 5. Upper bound communication complexity in $[L, U] = [1, n - 2]$.

B. Communication Complexity Analysis

The communication complexity of the proposed scheme is influenced by the total number of encrypted zeros and ones that are transmitted from the query user to the fog node and IoT nodes. We now examine the minimum and maximum number of encrypted blocks to obtain the lower and upper bounds of the communication cost.

Lower Bound Analysis: Consider a user's range query when the query is covering exactly half of the whole domain (as depicted in Fig. 4), namely, the left subtree of T ($[L, U] = [0, (n/2) - 1]$) or the right one ($[L, U] = [(n/2), n - 1]$). The length of the reduced path in both cases is one. Thus, the lower bound is achieved and the communication complexity is $O(1)$.

Upper Bound Analysis: The upper bound value can be obtained by issuing the range query: $[L, U] = [1, n - 2]$ (as shown in Fig. 5), in which the maximum number of encrypted blocks equals $(h + 1)(h + 2)$, where $h = \log_2 n$. Hence, the upper bound communication complexity is $O(\log^2 n)$

$$\begin{aligned}\text{Upper Bound} &: h + (h-1) + \cdots + 2 + 2 + \cdots + (h-1) + h \\ &= 2(h + (h-1) + (h-2) + \cdots + 2) \\ &= (h-1)(h+2) \in O\left(\log^2 n\right).\end{aligned}$$

C. Description of Our Proposed Scheme

In this section, we give a detailed description of the steps involved in our scheme.

1) Query User Key Generation: For simplicity, we consider $n = 2^h$ to represent n as an h-bit binary value. Given the input parameters k_0, k_1, and k_2, the query user generates a secret key $SK = (p, q, \mathcal{L})$ and a public parameter $PP = (k_0, k_1, k_2, \mathcal{N})$ for the SHE scheme. Then, the query user keeps the secret

key SK secretly and publishes the public parameter PP to the fog node and all IoT devices. It should be noted that achieving $O(\log^2 n)$ communication efficiency requires the following further assumptions.

1) Message space depends on k_1 and it should be set to at least $\lceil \log_2 N \rceil$, where N is the number of IoT devices, as it needs to cover the maximum value of the count range query $\text{COUNT}(\mathbf{I}') = |\mathbf{I}'|$, i.e., $|\mathbf{I}| = N$.
2) To achieve the intended communication efficiency, the SHE scheme should accept at least $2h$ homomorphic multiplication (h multiplication during XOR operation, $h-1$ for multiplying the XORs' outputs, and one multiplication by applying a filter to clear out dummy paths), i.e., k_0 should be set equal to $2(2h+1)k_2$ in order to successfully evaluate the $2h$-depth multiplicative circuit.

2) Range Query Generation at Query User: The range count query $\text{COUNT}(\mathbf{I}') = |\mathbf{I}'|$, where $\mathbf{I}' = \{I_i | I_i \in \mathbf{I} \wedge w_i \in [L, U]\}$, allows the user to retrieve the number of IoT devices I_i, whose data w_i are in the range $[L, U]$ where $0 \le L \le U \le n-1$. In order to fulfil the privacy-preserving range query, the following steps should be taken by the query user.

Step 1: Construct the PBTree T of height $h = \log_2 n$, with n leaf nodes labeled, from left to right, with $0, 1, \ldots, n-1$ and the left and right edges are labeled with zero and one, respectively.

Step 2: Generate the path from the root to leaves whose corresponding labels are in the range query, i.e., $t \in [L, U]$.

Step 3: Combine the path strings to obtain the reduced paths R. The length of each entry in R is between one and h, i.e., $\forall\ d \in R, 1 \le |d| \le h$.

Step 4: Invert each reduced path string by replacing zeros by ones and *vice versa*.

Step 5: Encrypt zeros and ones (separately) in each inverted path string by using SHE.

Step 6: Append dummy encrypted paths (encrypted zeros and ones) to prevent the fog node from inferring the actual query, more likely when it is a special-case query.

Step 7: Add the encrypted filter as a new column to wipe out the dummy paths (additional column in the final encrypted query in Fig. 2). More specifically, $E(1)$s are assigned to real reduced paths, whereas $E(0)$s are assigned to dummy paths, i.e., $E(0)$s behave as a filter to eliminate the dummy paths during the homomorphic aggregation. After that, the encrypted range query F will be sent to all IoT devices via the fog node. To be precise, the encrypted range query F includes the real paths, dummy paths, and the filtering column, together with an additional pair of encrypted zero and one, i.e., $(E(0), E(1))$. Note that each IoT device I_i can formulate its encrypted response without knowing the secret key SK during the response generation phase based on the additional pair of $(E(0), E(1))$.

3) Query Response at IoT Devices: Upon receiving the range query F, including real and additional dummy path strings, each IoT device $I_i \in \mathbf{I}$ needs to generate the encrypted response as follows. Since SHE is a symmetric encryption scheme, and no one other than the query user knows the secret key SK, each IoT device I_i cannot directly encrypt its data w_i. Therefore, instead of encrypting w_i, each IoT device can form the encrypted equivalent data $E(w_i)$ by combining $E(1)$'s and $E(0)$'s from the additional pair $(E(0), E(1))$ in the range query F.

Step 1: I_i converts the sensed and prepared data w_i into binary form, where $0 \le w_i \le n-1$. This is similar to traversing the PBTree with n leaves, from root to leaf node whose label is w_i. For example, as it is shown in Fig. 6, the path strings from root to leaves for two different sensed data 3 and 9 are "0011" and "1001," respectively.

Step 2: I_i generates the corresponding encrypted value for the binary representation of path string. I_i can generate the encrypted equivalent value for each sensed data w_i by combining $E(1)$'s and $E(0)$'s. For example, the encrypted equivalent value of "0011" and "1001" is $E(0011) = (E(0), E(0), E(1), E(1))$ and $E(1001) = (E(1), E(0), E(0), E(1))$.

Step 3: I_i performs the homomorphic XOR operations over all entries in the encrypted query F and previously computed $E(w_i)$ as listed in the large rectangles in Fig. 6.

Step 4: The output of the XORs are homomorphically multiplied to get the encrypted partial results. Then, these are multiplied by encrypted values in the supplementary filter column of F to clear out the dummy path strings from the response. For example, in the case of $w_i = 3$, our carefully devised filter column has nullified the effect of possible $E(1)$s from the response by setting the filter column to $E(0)$ for dummy path entries (yellow shaded rectangle while calculating the response for $w_i = 3$).

Step 5: The output of the previous step, in small rectangles, will homomorphically be added to each other to get the final encrypted response $Res[I_i]$

$$\text{Res}[I_i] = \sum_{\forall f \in F^{\text{real}}} E(1)\Big(\prod f \oplus w_i\Big) + \sum_{\forall f \in F^{\text{dummy}}} E(0)\Big(\prod f \oplus w_i\Big).$$

Step 6: Finally, to protect $\text{Res}[I_i]$ from the semihonest fog node, the IoT device I_i should self-blind the encrypted response $\text{Res}[I_i]$ by converting it into another valid ciphertext as follows:

$$\text{Res}[I_i] = \text{Res}[I_i] + (E(0) \times \mathsf{r})$$

where $\mathsf{r} \in \mathcal{M}$. This will prevent the fog node from generating different combinations of $E(0)$ and $E(1)$ of IoT device I_i to guess the corresponding sensed data w_i. Eventually, I_i can send back the self-blinded ciphertext result $Res[I_i]$ to the fog node.

4) Response Aggregation at Fog Node: After receiving all response values $Res[I_i]$ from IoT devices, the fog node aggregates the encrypted responses to generate the final encrypted result *Count* and forward it back to the user

$$\text{Count} = \sum_{I_i \in \mathbf{I}} \text{Res}[I_i].$$

5) Response Recovery at Query User: Upon receiving the encrypted query result *Count*, the query user uses secret key

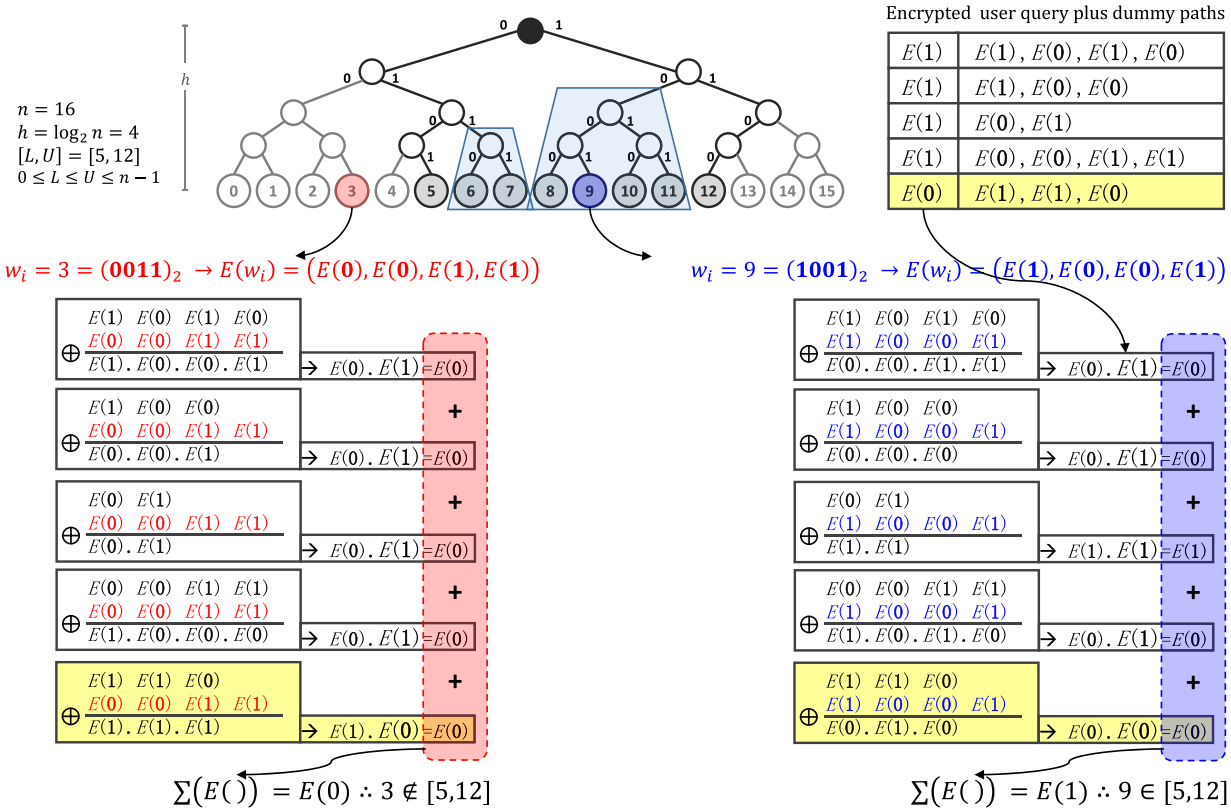


Fig. 6. IoT device I_i's response for two different sensed data $w_i = 3$ and 9, where $3 \notin [5, 12]$ and $9 \in [5, 12]$.

SK to recover the range query response as

$$\text{Count} \xrightarrow{D()} \text{Count}(\mathbf{I}') = |\mathbf{I}'| = \sum_{I_i \in \mathbf{I}} \text{Res}[I_i].$$

The correctness of the result is verified as follows:

$$\text{Count} = \sum_{I_i \in \mathbf{I}} \text{Res}[I_i] = \sum_{I_i \in \mathbf{I}'} E(1) + \sum_{I_i \notin \mathbf{I}'} E(0) = E(|\mathbf{I}'|).$$

V. Security Analysis

In this section, we analyze the security of our proposed scheme. We start with preserving the privacy of range query $[L, U]$ and then continue with protecting the privacy of subset $\mathbf{I}'$.

Query Range [L, U] Is Privacy Preserving in Our Proposed Scheme: As discussed in Section IV-A, in order to achieve communication efficiency, the query range $[L, U]$ $(0 \le L \le U \le n - 1)$ is turned into encrypted reduced paths F with minimum 2 and maximum $(\log n - 1)(\log n + 2)$ encrypted blocks. Each encrypted entry in F has been encrypted by SHE, which has been proved to be secure under the known-plaintext attack [17]. SHE's security is based on the large integer factorization problem (LIFP). Therefore, by properly setting the parameters, SHE is secure while LIFP is hard which means SHE guarantees that without knowing the secret key $SK = (p, q, \mathcal{L})$, the adversary has no idea of the entries inside F. Since both the fog device and IoT devices cannot access the secret key *SK*, they have no idea of the encrypted reduced path strings inside F, i.e., they cannot distinguish whether a ciphertext in F is encrypted from zero or one. Precisely, without plaintext information about entries in F, the fog node and IoT devices have no idea of the query range $[L, U]$ or even part of it.

Moreover, in the case of executing range queries with special values, e.g., $[L, U] = [0, n - 1]$, as F has just two single encrypted entries, i.e., $E(0)$ and $E(1)$, there is a possibility of inferring the range query. Hence, to prevent the fog node and IoT devices from guessing the range query, dummy paths are appended to the real entries in F as well as additional filter column to distinguish the real entries from dummy ones. Adding dummy path(s) provides an effective approach to reducing the probability of guessing the encrypted queries, especially when the query path length is short. For example, in Fig. 7, the length of the ciphertext query path (Q) is $\mathfrak{L} = 2$. That would mean that the curious entity can guess the original query by selecting one of the four possible cases. i.e., "00," "01," "10," or "11". Extending to the more general form, consider the original range query with one entry of length $\mathfrak{L}$. In this case, without injecting dummy path(s) there are $2^{\mathfrak{L}}$ different possible queries. Therefore, a curious entity can correctly guess the exact range query with probability $(1/2^{\mathfrak{L}})$. Injecting one dummy path instance of length $\mathfrak{L}'$ limits the probability of a correct guess by the adversary in two ways. First, since a dummy path instance of length $\mathfrak{L}'$ is injected, it generates $2^{\mathfrak{L}'}$ additional cases. Second, for the adversary, both a dummy path instance and a real range query path are indistinguishable; therefore, the adversary needs to search the whole string path of length $\mathfrak{L} + \mathfrak{L}'$ with $2^{\mathfrak{L}+\mathfrak{L}'}$ new cases. Consequently,

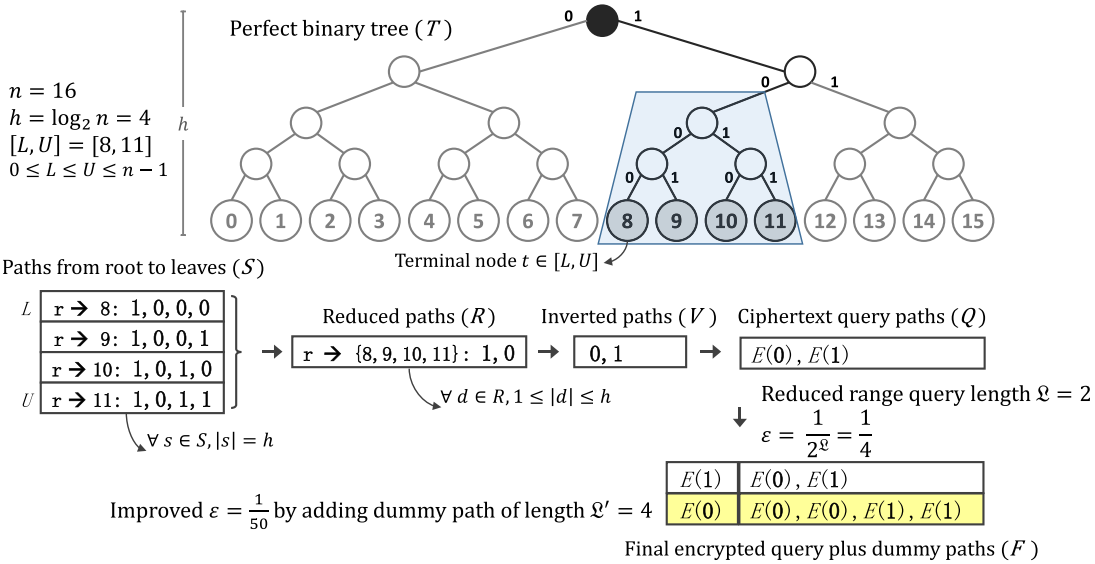


Fig. 7. Adding dummy path instance of length $\mathfrak{L}' = 4$ to original range query of length $\mathfrak{L} = 2$ to bind the probability (ε) of correctly guessing the user's range query; $\varepsilon = [1/(2^{\mathfrak{L}} + 2^{\mathfrak{L}'} + 2^{\mathfrak{L}+\mathfrak{L}'})]$.

a curious entity needs to guess the exact query from either dummy path instances ($2^{\mathfrak{L}'}$), real range query paths ($2^{\mathfrak{L}}$), or both ($2^{\mathfrak{L}+\mathfrak{L}'}$). This mean that the probability of guessing the user's real range query will be bounded from ($1/2^{\mathfrak{L}}$) to $[1/(2^{\mathfrak{L}} + 2^{\mathfrak{L}'} + 2^{\mathfrak{L}+\mathfrak{L}'})]$. Besides, to interdict the curious entity from successfully guessing the real range query and to bind the probability less than or equal to ε, we need to set $\mathfrak{L}' \ge \lceil \log_2 [((1/\varepsilon) - 2^{\mathfrak{L}})/(1 + 2^{\mathfrak{L}})] \rceil$, which is given as follows:

$$\frac{1}{2^{\mathfrak{L}} + 2^{\mathfrak{L}'} + 2^{\mathfrak{L}+\mathfrak{L}'}} \le \varepsilon \Rightarrow 2^{\mathfrak{L}} + 2^{\mathfrak{L}'} + 2^{\mathfrak{L}+\mathfrak{L}'} \ge \frac{1}{\varepsilon}$$
$$\Rightarrow 2^{\mathfrak{L}'}\left(1 + 2^{\mathfrak{L}}\right) \ge \frac{1}{\varepsilon} - 2^{\mathfrak{L}} \Rightarrow \mathfrak{L}'$$
$$\ge \left\lceil \log_2 \frac{\frac{1}{\varepsilon} - 2^{\mathfrak{L}}}{1 + 2^{\mathfrak{L}}}. \right\rceil.$$

For example, consider an original range query with an entry of length $\mathfrak{L} = 2$ in Fig. 7. In order to bind the probability of correctly guessing the range query from $\varepsilon = (1/2^{\mathfrak{L}}) = (1/4)$ to $\varepsilon = (1/50)$, we need to add dummy path(s) of length $\mathfrak{L}' \ge \lceil \log_2 [((1/\varepsilon) - 2^{\mathfrak{L}})/(1 + 2^{\mathfrak{L}})] \rceil = \lceil \log_2 [(50 - 2^2)/(1 + 2^2)] \rceil = 4$. Note that the dummy paths of length $\mathfrak{L}' > h$, where $h = \log_2 n$, need to be broken down into separate dummy path instances of maximum length h for each instance.

Subset $\mathbf{I}'$ Is Also Privacy Preserving in the Proposed Scheme: The subset $\mathbf{I}'$ denotes a set of IoT devices whose data are within the query range $[L, U]$, i.e., $\mathbf{I}' = \{I_i | I_i \in \mathbf{I} \wedge w_i \in [L, U]\}$. As described in the security model, $\mathbf{I}'$ should be kept secret from the query user, fog device, and each IoT device.

First, when the fog node's encrypted aggregated query response is received by the query user, he/she will decrypt using the secret key *SK* to obtain the final result. This means that the query user only knows the number of IoT devices whose data are within the requested query range $[L, U]$, but has no idea which specific IoT device has a data within the query range. Thus, the subset $\mathbf{I}'$ can be kept secret from the query user.

Second, on one hand, the fog node receives the encrypted query F from the query user and forward it to the IoT devices; on the other hand, the IoT devices' encrypted responses, i.e., $\text{Res}_{(I_i)}$s, will be aggregated at the fog node and the final ciphertext response $\sum_{I_i \in \mathbf{I}} \text{Res}[I_i]$ is forwarded back to the query user. Since the query range $[L, U]$ is transformed into encrypted query F by applying SHE, the security of SHE guarantees that the fog node has no idea on the plaintext of the query range. Then, the fog node has no way to determine the subset $\mathbf{I}'$ by observing encrypted path strings F. At the same time, each encrypted response $\text{Res}_{(I_i)}$ is self-blinded with random factor $(E(0) \times \mathsf{r})$ to make the brute-force attack impossible. This would mean that if there is no self-blind factor $(E(0) \times \mathsf{r})$, it would be possible for the fog node to identify the specific IoT device's response, i.e., it can iterate over all possible prepared values w_i $(0 \le w_i \le n-1)$ and compute the corresponding response for each entry in F to check whether $\text{Res}_{(I_i)} \stackrel{?}{=} \sum_{\forall f \in F^{\text{real}}} E(1)(\prod f \oplus w_i) + \sum_{\forall f \in F^{\text{dummy}}} E(0)(\prod f \oplus w_i)$. Although this attack requires a lot of computations, it is feasible, especially when n is small value. Luckily, protecting $\text{Res}[I_i]$ with $(E(0) \times \mathsf{r})$ will make it impossible for the fog node to launch a brute force attack on an individual IoT device's response. Therefore, it is impossible for the fog node to obtain the information about the subset $\mathbf{I}'$ and the subset $\mathbf{I}'$ is kept secret from the fog node.

Third, for each IoT device I_i, if it attempts to obtain the information about $\mathbf{I}'$, it needs to determine whether itself is in $\mathbf{I}'$ or not and whether other IoT devices are in $\mathbf{I}'$ or not. As discussed before, I_i uses its prepared data w_i to compute the encrypted $\text{Res}[I_i]$, but has no idea on $\text{Res}[I_i]$ is $E(0)$ or $E(1)$. In other words, I_i does not know whether w_i is in the query range $[L, U]$ and I_i is in $\mathbf{I}'$. Besides, since I_i cannot access the plaintext data prepared by the other IoT devices and the plaintext of query range, it has no idea whether the other IoT

TABLE II
PARAMETER SETTINGS

Shared parameter	Value
N	The number of IoT devices, $N = 1000$
n	The maximum sensed value, $n = \{2^6, 2^8, 2^{10}, 2^{12}, 2^{14}, 2^{16}, 2^{18}, 2^{20}, 2^{22}, 2^{24}, 2^{26}, 2^{28}, 2^{30}\}$
$[L, U]$	The range query lower and upper bound, $0 \leq L \leq U \leq n-1$
w_i	IoT device I_i's prepared data, randomly picked from $[0, n-1]$, i.e., $0 \leq w_i \leq n-1$

Our proposed and STSM scheme [17] based on SHE Cryptosystem	
Parameter	Value
h	$h = \log_2 n$, i.e., $h = \{6, 8, 10, \cdots, 30\}$
k_1, k_2	$k_1 = \log_2 N$, $k_2 = 40$
k_0	$2(h+1)k_2$ in [17], $2(2+1)k_2$ in our proposed scheme
p, q ,L, $\mathcal{N}$	$\lvert p\rvert = \lvert q\rvert = k_0$, $\lvert L\rvert = k_2$, $\mathcal{N} = pq$
$\mathcal{M}$	The query result $\mathcal{M} = N \in \{0,1\}^{k_1}$

Lu's scheme based on BGN Homomorphic Encryption	
Parameter	Value
h	$h = \sqrt{n}$, i.e., $h = \{2^3, 2^4, 2^5, \cdots, 2^{15}\}$
κ	$\kappa = 512$
p, q, $\mathcal{N}$	$\lvert p\rvert = \lvert q\rvert = \kappa = 512$, $\mathcal{N} = pq$
Δ	The upper bound of query result, $\Delta = N$

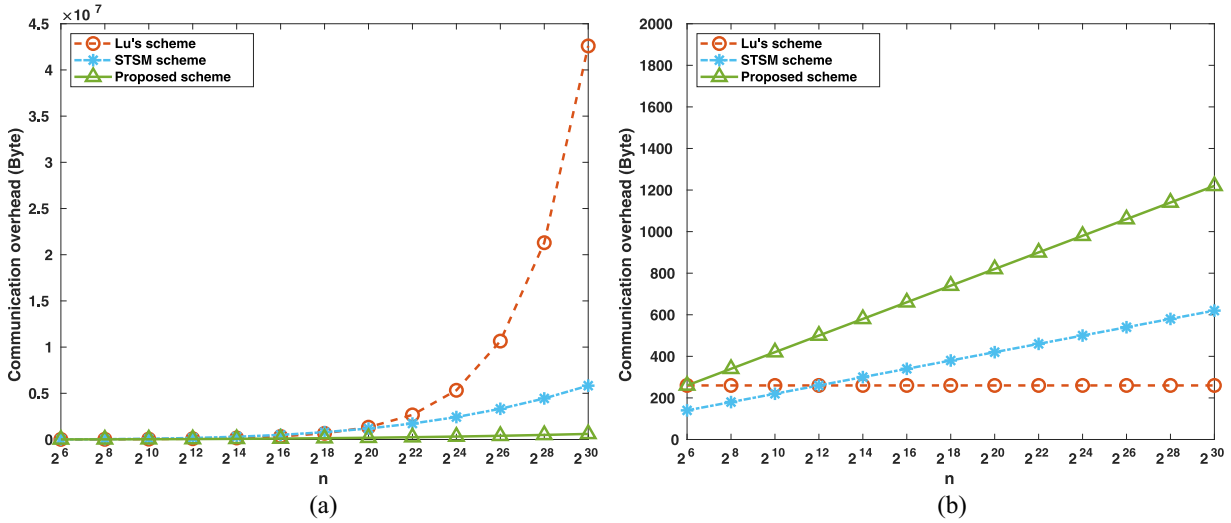


Fig. 8. Communication overhead comparisons (linear scale) between the proposed scheme and Lu's scheme with respect to n. (a) Requests from user to fog node. (b) Responses from fog node to user.

devices are in $\mathbf{I}'$ or not. Thus, the subset $\mathbf{I}'$ is kept secret from other IoT devices.

VI. PERFORMANCE EVALUATION

In this section, we give the result of an experimental assessment to analyze the effectiveness of the proposed scheme with respect to the computational costs and communication overheads. The detailed results confirm the practical performance of the scheme. Each step of the privacy-preserving range query in this scheme is compared with the corresponding steps in both previously proposed schemes, i.e., Lu's scheme [16] and the STSM scheme [17]. Lu's privacy-preserving range query, with $O(\sqrt{n})$ communication overhead, takes advantage of $\sqrt{n} \times \sqrt{n}$ matrix decomposition technique and is performed based on the BGN homomorphic cryptosystem [24]. The STSM scheme has a $O(\log^3 n)$ communication cost as it benefits from SHE as well as an encoding method to convert a given range query into $\log n + 1$ semitriangular sparse matrices. The platform used to compare the schemes was an Intel Core i5-2400 CPU @ 3.10 GHz, with 8 GB main memory running Linux (Ubuntu 16.04). The detailed parameter settings for both schemes and all experiments are listed in Table II.

A. Communication Overhead

For the communication overhead analysis, we report the required overhead in both directions between the user and the fog node, i.e., the volume of encrypted data that is uploaded from the query user to the fog node and the single ciphertext response length from the fog node to the query user. First, Fig. 8(a) compares the communication overhead of transferring an encrypted query from the query user to the fog node, with n varying from 2^{10} to 2^{30}. It is clear that the overhead of the Lu's scheme grows tremendously as n increases. This is mainly caused by the simultaneous increase in the number of encrypted elements. Our proposed scheme is also more efficient than the STSM scheme. Our $O(\log^2 n)$ communication pattern will make more efficient use of bandwidth as compared to Lu's scheme and the STSM scheme with $O(\sqrt{n})$ and $O(\log^3 n)$, respectively. For instance, in case of $n = 2^{30}$, the communication cost in our proposed scheme is less than one MB comparing to almost 40.6 and 5.5 MB in Lu's scheme and the STSM scheme, respectively. Therefore, the communication cost from the query user to the fog node in our proposed scheme is remarkably low.

Second, Fig. 8(b) depicts the encrypted response length from a fog node to the query user. All these three schemes are efficient and with responses being less than one KB. However,

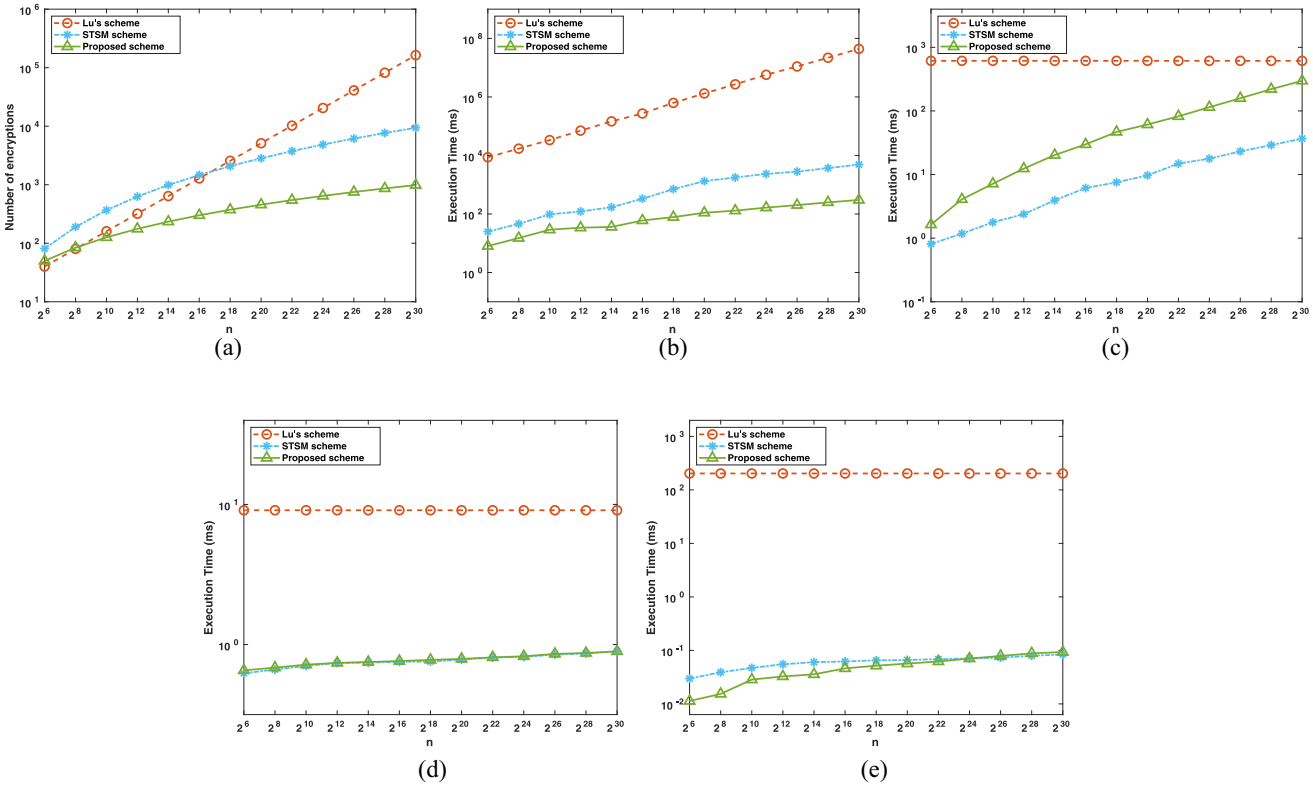


Fig. 9. Computational cost comparison between our scheme and Lu's scheme for n varying from 2^{10} to 2^{30}. (a) Number of encryptions in the decomposition phase. (b) Query generation at the user side. (c) Query response at the IoT device. (d) Response aggregation at the fog node. (e) Response recovery at the user side.

unlike the STSM scheme and our proposed scheme, Lu's scheme is irrelevant to n. This is due to the different values for k_0 that should be set to $2(2h+1)k_2$ and $2(h+1)k_2$ in our proposed scheme and the STSM scheme to support $2h$ and h-depth homomorphic multiplications, respectively. In Lu's scheme, it is based on the BGN, ciphertext length is constant and depends on the security parameter κ. For example, in the case of $\kappa = 512$, the BGN ciphertext size is $4\kappa = 256$ B or 0.25 KB.

B. Computational Cost

As well, in all three schemes, the computational costs are influenced by the number of encrypted elements that are generated by the query user during the range query encoding phase. Fig. 9(a) compares the number of encrypted in all three schemes. The computational costs of Lu's scheme and the STSM scheme increase rapidly with n. The difference is rooted in the decomposition method. Lu's scheme and the STSM scheme use $O(\sqrt{n})$ and $O(\log^3 n)$ decomposition approaches, respectively, whereas ours take advantage of a faster and more efficient decomposition technique with $O(\log^2 n)$ communication efficiency.

Note: Execution times for Lu's scheme are very high compared with both the STSM scheme and our proposed scheme (almost 1 ms in three out of the four subfigures). We have, therefore, used a logarithmic scale to make interpretation easier.

Fig. 9(b) shows the consuming time for query generation by the user. From the figure, we see that the computational cost of Lu's scheme dramatically grows with n while the STSM and our proposed schemes have a significantly lower growth rate. We also observe that our scheme has a better execution time than the STSM scheme. The reason behind this is that the schemes are producing a different number of encrypted elements as discussed before. Therefore, our proposed scheme achieves better results than the others in terms of query generation time.

Fig. 9(c) illustrates the response time for and IoT device. Although the IoT device response time in Lu's scheme is unaffected by problem size, it is still considerably higher than the other two. Additionally, we observe that our proposed scheme requires more computational time than the STSM scheme, as the IoT device's response generation needs more computation, i.e., a combination of XOR operations as well as more homomorphic additions and multiplications than the STSM scheme. However, the response time in both schemes is much less than 1 s. Putting these two results together, our proposed scheme's query generation time is 16 times faster than that of the STSM scheme, though the response time for each IoT device in our scheme is slightly longer than that of the STSM scheme. As a result, the overall execution time in our proposed scheme is better than the STSM scheme.

Fig. 9(d) depicts the response aggregation time at the fog node. It is mainly dependent on the number of IoT devices, i.e., N. Therefore, as it is seen from the figure, the aggregation time in all three schemes is independent of the problem size, i.e., n, even if there is a slight increase in both our proposed and the STSM schemes, varying with n. This is because of the

gradually increasing ciphertext sizes with respect to problem size. In Lu's scheme, the ciphertext length depends only on the security parameter κ and is consequently independent of n. Moreover, the aggregation times in both our proposal and the STSM scheme are almost the same as both schemes employ the SHE cryptosystem.

Finally, Fig. 9(e) presents the response recovery time that measures how much time the user spends decrypting the ciphertext upon receiving the encrypted response from the fog node. Obviously, they are still efficient and independent of problem size n. However, decryption in both the proposed and the STSM schemes (with a time less than one millisecond even for $n = 2^{30}$) is faster than Lu's scheme. The reason behind this is that the latter performs in the BGN scheme and over $\mathbb{G}_T$ as well as requiring relatively smaller message space than $\Delta = 1000$ to operate more effectively. Message decryption in the BGN scheme involves solving the discrete logarithm using Pollard's lambda method, i.e., the BGN scheme is time consuming and less practical for larger message spaces.

It should also be noted that in addition to improving significant portions of the query processing systems, our proposed scheme has addressed two limitations of the other two schemes, namely: 1) ours does not require the lower and upper bound values be a power of two whereas the STSM scheme does and 2) ours support noncontinuous range queries while theirs do not.

VII. Related Work

In this section, we will take a brief look at some previously reported studies in privacy-preserving schemes for fog-based IoT applications. Most recently, Mahdikhani *et al.* [17] have devised the SHE scheme along with decomposition technique to transform the range queries into semitriangular sparse matrices to achieve $O(\log^3 n)$ communication efficiency. Although the STSM scheme has acceptable communication and computation efficiency, it only accepts continuous range queries in the form of $[L = 2^a, U = 2^{a'}]$, i.e., the lower and upper bound values should be chosen as a power of two. Prior to this study, a privacy-preserving range query in fog-based IoT was studied in [16], which is another related study to our proposed scheme. The scheme proposed in [16] is built upon the BGN homomorphic encryption together with a range query expression, decomposition, and composition technique, which can achieve $O(\sqrt{n})$ communication efficiency. It is less efficient than the STSM scheme, but it accepts any continuous form of range queries. In addition, because the scheme is based on the BGN scheme with time-consuming bilinear pairing operations in public-key settings, the computational cost is large. Aiming at improving the communication efficiency as well as resolving the deficiencies in the queries' form, our proposed scheme comes with a novel range query encoding technique to achieve $O(\log^2 n)$ communication efficiency and overcome the weakness of the query format.

When the privacy-preserving range query on the number of IoT devices whose data x_i is within the range [L,U] is considered as a special privacy-preserving data aggregation scheme in fog-based IoT, there are other studies close to our proposed scheme. Lu *et al.* [13] addressed heterogeneous data aggregation in real IoT applications by proposing a lightweight privacy-preserving data aggregation (LDPA) for a fog-enabled setting. The proposed LPDA is characterized by applying Paillier cryptosystem, the Chinese Remainder Theorem, and one-way hash chain function to aggregate hybrid IoT devices' data and to early filter the injected false data at the network edge. Huang *et al.* [25] studied the fog-assisted selective aggregation operation. Specifically, they constructed a new threat model to formalize the noncollusive and collusive attacks of compromised fog nodes. Their proposed privacy preserving and reliable selective multisource aggregation scheme is comprised of the BCP cryptosystem, randomized message-lock encryption, homomorphic proxy-authenticators, and multidimensional aggregation and can well tackle the data privacy and reliability challenges. Recently, Mahdikhani *et al.* [26] presented a privacy-preserving subset aggregation scheme in fog-enhanced IoT scenarios, which enables a query user to gain the sum of the prepared data from a subset of IoT devices. To identify the subset, the inner product similarity of the normalized vectors in the query user side and each IoT device is securely computed. Only when the inner product is greater than the user's specified threshold, an IoT device's data will be privately aggregated to form the final response.

VIII. Conclusion

In this article, we have proposed a novel privacy-preserving range query encoding technique based on SHE homomorphic encryption [17] and reduced paths concept to privately and efficiently execute range queries with $O(\log^2 n)$ communication efficiency. Compared with previously discussed studies [16], [17], our proposed scheme can also support noncontinuous range queries for any given value of L and U. Also, extensive experimental results demonstrate that our proposed scheme shows significant improvements in both communication overhead and computational cost. Consequently, our proposed scheme is more efficient privacy-preserving range query in fog-based IoT environments. In the future, we plan to expand this study by launching a more complex configuration in the system model.

References

[1] Q. Kong, R. Lu, M. Ma, and H. Bao, "A privacy-preserving and verifiable querying scheme in vehicular fog data dissemination," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1877–1887, Oct. 2018.

[2] Q. Kong, R. Lu, M. Ma, and H. Bao, "A privacy-preserving sensory data sharing scheme in Internet of Vehicles," *Future Gener. Comput. Syst.*, vol. 92, pp. 644–655, Mar. 2019.

[3] M. Li, L. Zhu, Z. Zhang, X. Du, and M. Guizani, "PROS: A privacy-preserving route-sharing service via vehicular fog computing," *IEEE Access*, vol. 6, pp. 66188–66197, 2018.

[4] Y.-D. Chen, M. Z. Azhari, and J.-S. Leu, "Design and implementation of a power consumption management system for smart home over fog-cloud computing," in *Proc. IEEE 3rd Int. Conf. Intell. Green Build. Smart Grid (IGBSG)*, 2018, pp. 1–5.

[5] X. Li, R. Lu, X. Liang, X. Shen, J. Chen, and X. Lin, "Smart community: An Internet of Things application," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 68–75, Nov. 2011.

[6] Y. Yang, X. Luo, X. Chu, and M.-T. Zhou, "Fog-enabled smart home and user behavior recognition," in *Fog-Enabled Intelligent IoT Systems*. Cham, Switzerland: Springer, 2020, pp. 185–210.

[7] H. Bao and R. Lu, "A new differentially private data aggregation with fault tolerance for smart grid communications," *IEEE Internet Things J.*, vol. 2, no. 3, pp. 248–258, Sep. 2015.
[8] H. Bao and L. Chen, "A lightweight privacy-preserving scheme with data integrity for smart grid communications," *Concurrency Comput. Pract. Exp.*, vol. 28, no. 4, pp. 1094–1110, 2016.
[9] H. Bao and R. Lu, "A lightweight data aggregation scheme achieving privacy preservation and data integrity with differential privacy and fault tolerance," *Peer-to-Peer Netw. Appl.*, vol. 10, no. 1, pp. 106–121, 2017.
[10] N. Saxena, B. J. Choi, and R. Lu, "Authentication and authorization scheme for various user roles and devices in smart grid," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 5, pp. 907–921, Feb. 2016.
[11] M. A. Salahuddin, A. Al-Fuqaha, M. Guizani, K. Shuaib, and F. Sallabi, "Softwarization of Internet of Things infrastructure for secure and smart healthcare," 2018. [Online]. Available: arXiv:1805.11011.
[12] M. Yang, T. Zhu, B. Liu, Y. Xiang, and W. Zhou, "Machine learning differential privacy with multifunctional aggregation in a fog computing architecture," *IEEE Access*, vol. 6, pp. 17119–17129, 2018.
[13] 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.
[14] H. Bao, R. Lu, B. Li, and R. Deng, "BLITHE: Behavior rule-based insider threat detection for smart grid," *IEEE Internet Things J.*, vol. 3, no. 2, pp. 190–205, Jul. 2015.
[15] M. A. Ferrag, A. Derhab, L. Maglaras, M. Mukherjee, and H. Janicke, "Privacy-preserving schemes for fog-based IoT applications: Threat models, solutions, and challenges," in *Proc. IEEE Int. Conf. Smart Commun. Netw. Technol. (SaCoNeT)*, 2018, pp. 37–42.
[16] R. Lu, "A new communication-efficient privacy-preserving range query scheme in fog-enhanced IoT," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2497–2505, Feb. 2020.
[17] H. Mahdikhani, R. Lu, Y. Zheng, J. Shao, and A. Ghorbani, "Achieving $O(\log^3 n)$ communication-efficient privacy-preserving range query in fog-based IoT," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5220–5232, Jun. 2020.
[18] K. Renuka, S. N. Das, and K. H. Reddy, "An efficient context management approach for IoT," *IUP J. Inf. Technol.*, vol. 14, no. 2, pp. 24–35, 2018.
[19] H. Liang, J. Wu, Z. Liu, and J. Li, "Shape-unconstrained privacy-preserving range query for fog computing supported vehicular networks using image," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, 2019, pp. 683–688.
[20] M. Li, L. Zhu, and X. Lin, "Efficient and privacy-preserving carpooling using blockchain-assisted vehicular fog computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4573–4584, Aug. 2018.
[21] K. Lewi and D. J. Wu, "Order-revealing encryption: New constructions, applications, and lower bounds," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 1167–1178.
[22] X. Yuan, X. Wang, C. Wang, B. Li, and X. Jia, "Enabling encrypted rich queries in distributed key-value stores," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 6, pp. 1283–1297, Jun. 2018.
[23] Q. Wang *et al.*, "Searchable encryption over feature-rich data," *IEEE Trans. Depend. Secure Comput.*, vol. 15, no. 3, pp. 496–510, Jun. 2018.
[24] D. Boneh, E. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts," in *Proc. TCC*, Cambridge, MA, USA, Feb. 2005, pp. 325–341.
[25] C. Huang, D. Liu, J. Ni, R. Lu, and X. Shen, "Reliable and privacy-preserving selective data aggregation for fog-based IoT," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.
[26] H. Mahdikhani, S. Mahdavifar, R. Lu, H. Zhu, and A. A. Ghorbani, "Achieving privacy-preserving subset aggregation in fog-enhanced IoT," *IEEE Access*, vol. 7, pp. 184438–184447, 2019.

Hassan Mahdikhani (Graduate Student Member, IEEE) received the B.Eng. degree in computer engineering-software from Kharazmi University, Tehran, Iran, in 2001, and the M.Eng. degree in computer engineering-software from Iran University of Science and Technology, Tehran, in 2006. He is currently pursuing the Ph.D. degree in computer science with the University of New Brunswick (UNB), Fredericton, NB, Canada.

He is a Cybersecurity Researcher with the Canadian Institute for Cybersecurity, UNB. His research interests include cloud computing security, secure and privacy-preserving computation offloading, and applied cryptography.

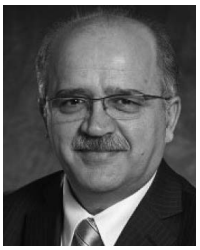
Rongxing Lu (Senior Member, IEEE) received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012.

He is an Associate Professor with the Faculty of Computer Science, University of New Brunswick (UNB), Fredericton, NB, Canada.

Dr. Lu was awarded the most prestigious "Governor General's Gold Medal," from the Department of Electrical and Computer Engineering, University of Waterloo in 2012 and won the eighth IEEE Communications Society Asia Pacific Outstanding Young Researcher Award in 2013. He is the Winner of Excellence in Teaching Award, FCS, UNB from 2016 to 2017. He currently serves as the Vice-Chair (Conferences) of IEEE ComSoc CIS-TC. He is currently a Senior Member of IEEE Communications Society.

Jun Shao received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2008.

He was a Postdoctoral Fellow with the School of Information Sciences and Technology, Pennsylvania State University, State College, PA, USA, from 2008 to 2010. He is currently a Professor with the School of Computer Science and Information Engineering, Zhejiang Gongshang University, Hangzhou, China. His current research interests include network security and applied cryptography.



Ali A. Ghorbani (Senior Member, IEEE) received the M.Sc. degree in computer science from George Washington University, Washington, DC, USA, in 1979, and the Ph.D. degree in computer science from the University of New Brunswick, Fredericton, NB, Canada, in 1995.

He has held a variety of positions in academic for the past 37 years is currently a Professor of Computer Science with the Tier 1 Canadian Institute for Cybersecurity, the Director of the Canadian Institute for Cybersecurity, University of New Brunswick, which he established in 2016, and an IBM Canada Faculty Fellow, where he is also the Founding Director of the Laboratory for Intelligence and Adaptive Systems Research. He is a co-inventor on three awarded patents in the area of Network Security and Web Intelligence and has published over 270 peer-reviewed articles during his career. His current research focus is cybersecurity, webintelligence, and critical infrastructure protection.

Prof. Ghorbani served as the Co-Editor-In-Chief of *Computational Intelligence: An International Journal* from 2007 to 2017. He is the Co-Founder of the Privacy, Security, Trust Network in Canada and its international annual conference. He has supervised over 180 research associates, postdoctoral fellows, graduate and undergraduate students during his career.