A Secure Collusion-aware and Probability-aware Range Query Processing in Tiered Sensor Networks

Lei Dong[†‡], Xuan Chen[‡], Jianxiang Zhu[†‡], Hong Chen*[†‡], Ke Wang[§], Cuiping Li[†‡]
[†]Key Laboratory of Data Engineering and Knowledge Engineering of Ministry of Education, Beijing, China
[‡]School of Information, Renmin University of China, Beijing, China
[§]School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
* Corresponding Author. Email: chong@ruc.edu.cn

***Abstract*—With high expansibility and efficient power usage, tiered wireless sensor networks are widely deployed in many fields as an important part of Internet of Things (IoTs). It is challenging to process range query while protecting sensitive data from adversaries. Moreover, most existing work focuses on privacy-preserving range query neglecting collusion attacks and probability attacks, which are more threatening to network security. In this paper, we first propose a secure range query protocol called secRQ, which not only protects data privacy, but also resists collusion attacks and probability attacks in tiered wireless sensor networks. Generalized inverse matrices and distance-based range query mechanism are used to guarantee security as well as high efficiency and accuracy. Besides, we propose the *mutual verification* scheme to verify the integrity of query results. Finally, both theoretical analysis and experimental results confirm the security, efficiency and accuracy of secRQ.**

***Keywords*—Range query, Wireless sensor networks, Privacy preservation, Collusion attack, Probability attack, Integrity verification**

I. Introduction

As an indispensable building part of Internet of Things (IoTs), wireless sensor network (WSN) has been widely used in many applications, including smart home, e-health and environment monitoring. Most existing large-scale WSNs follow a tiered architecture [1] as shown in Fig.1, which consists of a large amount of resource-limited sensor nodes in the lower tier and several resource-rich storage nodes in the upper tier. Sensor nodes are mainly responsible for sensing, while storage nodes store data received from sensor nodes and answer queries received from the sink. There are several advantages of the tiered WSNs. On the one hand, sensor nodes can save much energy and avoid communication bottleneck by sending data to their closest storage nodes instead of the sink [2], meanwhile, it can also reduce storage cost. On the other hand, the query processing becomes more efficient since the sink only needs to communicate with several storage nodes rather than a large amount of sensor nodes [3]. Therefore, the tiered architecture is favorable for extending capacity and improving scalability of WSNs.

However, this architecture also brings in some security risks. Since storage nodes are responsible for storing data and processing queries, they are more liable to be compromised. Once compromising a storage node, the adversary can obtain all sensitive data under an unencrypted format, which seriously violates data privacy. Moreover, the adversary can manipulate the compromised storage node to submit damaged (*e.g.* fake or incomplete) results, which breaches result integrity. As a result, it is challenging to preserve data privacy and result integrity as well as achieving efficient performance and precise results.

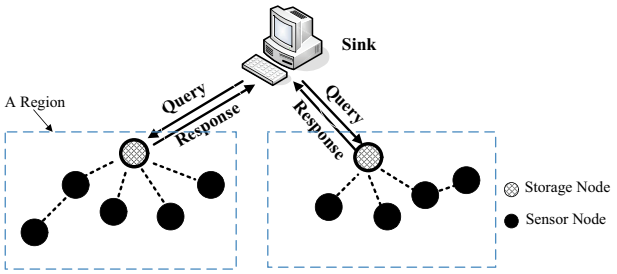


Fig. 1: Architecture of Tiered WSNs

Range query is one of the most important queries in WSNs. A privacy-preserving range query processing protocol should protect sensitive data and actual queries from adversaries. For example, considering the privacy requirements for patients in e-health, each wearable health monitor device is a sensor node to collect the patient's physical signs, such as heart rate, blood pressure, body temperature. Due to the limitation of power and storage capacity, these health monitors send the physical signs to the medical storage nodes at the specified epoch. When the doctor intends to know the blood pressure information of tachycardia patients whose normal heart rates are between 100bpm and 120bpm, he can propose a query range [100bpm, 120bpm] to the sink, and the sink will send the range query to the medical storage nodes. In this case, two crucial items are expected to be protected from adversaries. One is sensitive data, *e.g.* the patients' physical signs sensed by health monitors (treated as data privacy of sensor nodes), and the other is actual query, *e.g.* the query range [100bpm, 120bpm] (treated as data privacy of users) proposed by the doctor. If the adversaries capture the query range, at least they can know (1) what the doctor concerns about (2) the physical signs in the result corresponds to persons whose heart rates are between 100bmp and 120bmp. Therefore, sensitive data and actual queries should not be transmitted explicitly.

To defend attacks, the conventional way is to apply encryption on sensitive data and to construct characteristic values (*e.g.* data range prefix [4], Bloom filter code [5]) for sensitive data and actual queries. Storage nodes can process range queries on these characteristic values without knowing sensitive data and actual queries. Most existing secure range query protocols

[4]–[9] only study privacy preservation when storage nodes are compromised. Work [10] first focused on collusion attacks for range queries in tiered WSNs, but it did not consider probability attacks. In fact, collusion attacks [11] and probability attacks [12] are becoming increasingly common in WSNs.

Collusion attack means that the adversary captures many nodes. Consider a special situation where the adversary captures some sensor nodes and their storage node. The adversary can deduce more information to break the network security according to algorithms in sensor nodes and stored data items in storage node. For instance, the adversary compromises one health monitor and the corresponding medical storage node, gets the characteristic value construction of heart rate from the compromised health monitor, and then recovers the heart rate of other patients stored in the compromised medical storage nodes. Probability attacks in WSNs are based on (1) historical or prior knowledge, and (2) the uniform one-to-one privacy-preserving characteristic value construction, leading to that each data item has a unique fixed characteristic value. The adversary can not only find out the corresponding relationships between data (or query) and their characteristic value by means of probability matching, but also exploit pairs of sensitive data item and its characteristic value to crack character-generated function to recover other data items. Due to the historical knowledge that the doctor constantly concerns about the tachycardia patients whose heart rates are between 100bpm and 120bpm, the adversary can decide which characteristic value is most likely to represent [100bpm, 120bpm] according to frequency statistics in the compromised medical storage node.

A robust range query protocol in WSNs should consider as much kinds of attack as possible to enhance resistance under the premise of high efficiency and accuracy. Besides, integrity verification mechanism also becomes a hot research topic. Thus, how to achieve high precision and efficient performance while resisting to adversaries even under collusion attacks and probability attacks, and how to design an efficient integrity verification to detect damaged results, are significant for the development of WSNs. In this paper, we propose a secure range query processing protocol called secRQ in tiered WSNs. Our secRQ can resist the typical and risky attacks in WSNs, such as plaintext attacks, collusion attacks and probability attacks. To the best of our knowledge, this paper is the first to consider collusion attacks as well as probability attacks for range queries in tiered WSNs. Our secRQ consumes less communication cost than the latest work [10], and achieves 100% accuracy. Furthermore, we propose the *mutual verification* scheme that allows the sink to verify the integrity of query results. The *mutual verification* scheme achieves stronger scalability than existing prior integrity verification schemes [4] and [9] for range queries in tiered WSNs.

The rest of paper is organized as follows. Section II summarizes the related work. Models and goals of secure range query are described in Section III. Section IV introduces our basic secRQ protocol for 1-dimensional data. We extend the basic secRQ for multi-dimensional data in Section V. The *mutual verification* scheme for integrity verification is elaborated in Section VI. We analyze secRQ in terms of complexity, privacy and integrity in Section VII. Section VIII evaluates secRQ. We conclude this paper in Section IX.

II. Related work

The problem of privacy and integrity preserving range query in tiered WSNs has been investigated in work [4]–[10]. Sheng *et al.* proposed a verifiable privacy-preserving range query protocol in tiered WSNs [6]. The protocol divides the domain of data value into multiple disjoint buckets. When a range query comes, the sink converts the query to a set of bucket IDs covering the required range and sends the IDs to storage nodes, then storage nodes returns the encrypted data items in these buckets to the sink. This bucket technique may bring in a large number of false positives and cause high communication cost. It can be hardly extended to solve multi-dimensional range query.

To address this problem, Chen *et al.* presented SafeQ, a secure and efficient range query processing protocol [4], [7]. It uses prefix membership verification scheme [13] to represent each sensed data ranges, and then utilizes the method in [14] to ensure data privacy and puts forward neighborhood chain to protect integrity. However, it demands much computation and communication.

Zhang *et al.* proposed ESRQ, an efficient and secure range query protocol for 1-dimensional range query in tiered WSNs [5]. ESRQ utilizes Bloom filter technique to encode sensed data items and queries, and processes range queries according to Bloom filter membership test. Li *et al.* proposed BBP, a broadcast-based protocol for multi-dimensional range query in sensor network [8]. It constructs an individual Bloom filter for each dimension. The storage nodes in BBP are credible and regarded as converters to form different ranges for different dimensions. Due to false positives brought by Bloom filter, both ESRQ and BBP cannot provide precise query results.

Yi *et al.* proposed an order preserving function-based protocol [9]. The protocol uses two monotone increasing functions $pf(x, i)$ and $Qf(x, y)$ to encode sensor data items and issued range queries respectively. This protocol cannot defend against node collusion. The reason is that $pf(x, i)$ and $Qf(x, y)$ share the same main body, and the degree of $pf(x, i)$ cannot be large due to limitation of sensor node's ability.

Collusion attack was first considered in work [10]. Confusion-based collusion-aware privacy-preserving range query protocol (c-CPRQ) and l-uncertainty collusion-aware privacy-preserving range query (u-CPRQ), were proposed to process privacy-preserving range query based on Bloom filter. The basic idea of these protocols is that different sensor nodes have different hash functions to encode data items, which prevents sensor nodes from deducing data items of other nodes. The storage node processes range queries according to Bloom filter membership test. c-CPRQ may generates unallowed false negatives, which means the storage may filter out some data items that are actually in the range. u-CPRQ overcomes this shortage of c-CPRQ while producing more false positives.

In existing secure range query protocols, data integrity has also attracted much attention. Multi-dimensional neighborhood chains [4] are used to preserve the integrity of multi-dimensional data, but it needs to be assisted by the vulnerable and unbelievable storage nodes. Yi *et al.* [9] proposed a link watermarking scheme to check 1-dimensional data integrity and designed a data structure called multi-dimensional

neighbor tree (MDNT) to validate the integrity for multi-dimensional data. Nevertheless, MDNT excessively increases communication overhead and energy consumption, and it is built in storage nodes, which are not reliable. Chen *et al.* put forward a new data structure called local bit matrices to detect the damaged results with high probability [15] in cloud computing, but the structure overly depends on the data reasonable distribution. Only when the data distribution draws on the partition buckets, can the bit matrices accurately examine the loss of data.

III. Models and Problem Statement

A. Network Model

We consider a typical tiered sensor network as shown in Fig.1. Copious resource-limited sensor nodes are responsible for sensing data in the lower layer. Storage nodes in the upper layer store data received from sensor nodes, process queries received from the sink and reply results to the sink. The sink interacts with users and issues queries to the storage nodes. The network is partitioned into several regions. Each region contains a storage node and some sensor nodes. When the sink receives a query from a user, it transforms the query to a new format and then sends it to storage nodes in the specified regions. While receiving the transformed query, storage nodes process the query over stored data and return the requested data satisfying the range query to the sink. Finally the sink checks the results and replies data to the user.

B. Adversary Model

We assume that the adversary aims at obtaining sensitive data of sensor nodes and actual queries of users. According to work [4]–[10], the sink is reliable. Oppositely, sensor nodes and storage nodes are unreliable. A single compromised sensor node only leaks its own data items while a compromised storage nodes may leak more information (*e.g.* stored data items received from sensor nodes and queries received from the sink). What is worse, a compromised storage node may be manipulated to reply fake or incomplete results to the sink. Furthermore, it is worth noting two harmful attacks. One is collusion attack, where a compromised storage node cooperates with one or more compromised sensor nodes to steal more information. More specifically, if all sensor nodes in the same region share unified parameters of the query protocol, the adversary can have high level knowledge, (*e.g.*, algorithm, parameters) of the query processing mechanism and the privacy-preserving scheme through compromised sensor nodes, and can use the knowledge to recover sensitive data items stored in the storage node. The other is probability attack. The adversary guesses the true values with high probabilities according to the additional knowledge *e.g.* common sense and distribution frequencies and then gets the characteristic values and recovers sensitive data. In this paper, we mainly focus on (1) preventing sensitive data items and actual queries from compromised storage nodes, (2) resisting collusion attacks and probability attacks and (3) detecting fake and incomplete results.

C. Design Requirements

The fundamental problem in this paper is: *how to design the range query processing protocol in a privacy-integrity-preserving manner that can resist collusion attacks and probability attacks?* The protocol should achieve the following goals.

- Query and data privacy. Query privacy means that the storage nodes cannot derive the actual query range. Similarly, data privacy means that the actual data items sensed by sensor node S should only be learned by S and the sink under collusion attacks. In addition, the scheme should prevent the adversary from one-to-one mapping relationship between characteristic values and sensitive data items and that between transformed queries and actual queries.
- Query process efficiency and accuracy. Energy is the bottleneck of sensor network development. Work in [16] indicates that communication consumes much more energy than other operations. Hence, the less communication overhead, the higher efficiency. Besides, accuracy is an important quality metric of range query protocols. The higher accuracy, the better protocol.
- Result integrity. Since a compromised storage node may reply fake or incomplete results, it is necessary to design an integrity verification mechanism for the sink to detect damaged results.

D. Notations

To make our paper easier to follow, we first summarize the main notations used in our paper as shown in Table.I.

Table I: Notations

Notation	Meaning
S_i	a sensor node with unique ID i
ST	a storage node
t	an epoch
T	the user-specified time segment
z	the number of attribute dimensions
$d_{i,j}/\ D_{i,j}$	the j-th 1-dimensional/multi-dimensional data item collected by S_i
k_i	a secret key of S_i, $()_{k_i}$ means an encryption function using k_i
W_s	a 3×2 matrix generated by the sink
$M_s/\ M_i$	generalized inverse matrix(matrices) owned by the sink/S_i
$v_{i,j}/\ V_{i,j}$	data vector of $d_{i,j}/\ datamatrix of D_{i,j}$
$cv_{i,j}/\ CV_{i,j}$	characteristic value of $d_{i,j}/\ D_{i,j}$
$R/\ R_z$	1-dimensional query range / z-dimensional query range
a, b	low bound and upper bound of R
r_{mid}, r_{bou}	$r_{mid} = (a+b)/2$, $r_{bou} = a$ or b
$q_{mid}/\ Q_{mid}$	query middle-value vector of R/ query middle-value matrix of R_z
$q_{bou}/\ Q_{bou}$	query bound vector of R/ query bound matrix of R_z
$tr_{mid}/\ TR_{mid}$	transformed query middle-value vector/matrix
$tr_{bou}/\ TR_{bou}$	transformed query bound vector/matrix
$MV_{i,j}$	mutual verification object of $d_{i,j}$ or $D_{i,j}$
RS	whole result set of range query
RS_i	result subset of of range query from S_i, $RS_i \in RS$
DS	plaintext data set that comes from RS
VS	mutual verification object set of DS
CS	combined space used in *mutual verification* scheme

IV. Privacy-preserving Range Query Processing for 1-dimensional Data

In this section, we elaborate our basic privacy-preserving range query protocol secRQ for 1-dimensional data. Then we will extend this basic model for multi-dimensional data in Section V.

A. Overview

Our basic range query protocol is based on the distance comparison rule. The distance-based query model is described as follows. For 1-dimensional range $R = [a, b]$, we use $\langle r_{mid}, r_{bou} \rangle$ to represent R, where r_{mid} denotes the middle point value of R, that is $r_{mid} = (a + b)/2$, and r_{bou} denotes an arbitrary bound of R, that is $r_{bou} = a$ or $r_{bou} = b$. Fig.2 shows the idea of distance comparison.

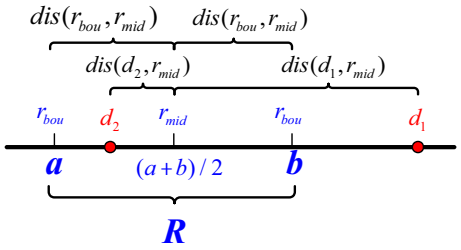


Fig. 2: Distance Comparison

Theorem 1. *(Distance Comparison Rule) For a 1-dimensional range $R = [a, b]$, let $r_{mid} = (a + b)/2$ and $r_{bou} = a$. A 1-dimensional data item d is in R if and only if $dis(d, r_{mid}) \leqslant dis(r_{bou}, r_{mid})$, where $dis()$ means the Euclidean distance.*

In Fig.2, we can decide d_1 is out of R according to $dis(d_1, r_{mid}) > dis(r_{bou}, r_{mid})$, and d_2 is in R according to $dis(d_2, r_{mid}) < dis(r_{bou}, r_{mid})$. Note that two data items cannot be recovered only by their Euclidean distance. Thus, in some ways, we can determine whether d is in R through *Distance Comparison Rule* in a privacy-preserving method.

The basic idea of our 1-dimensional privacy-preserving range query processing is detailed as follows. We assume that in the network of N sensor nodes, the sensor node S_i $(1 \leqslant i \leqslant N)$ senses n data items $d_{i,1}, \ldots, d_{i,n}$ at epoch t (In view of limitation of sensor nodes' energy, the epoch is much larger than sensing interval, *i.e*, $n > 1$). In order to preserve data privacy, S_i encrypts $d_{i,1}, \ldots, d_{i,n}$ using its secret key k_i and then obtains $(d_{i,1})_{k_i}, \ldots, (d_{i,n})_{k_i}$. Meanwhile, S_i constructs characteristic values $cv_{i,j}$ for each data item $d_{i,j}$ $(1 \leqslant j \leqslant n)$. The message that S_i sends to storage node ST consists of four parts: sensor node ID i, epoch t, encrypted data items $(d_{i,1})_{k_i}, \ldots, (d_{i,n})_{k_i}$, and $cv_{i,1}, \ldots, cv_{i,n}$. While receiving a range query $\langle T, R \rangle$, to preserve the users privacy, the sink first computes two parameters $\langle r_{mid}, r_{bou} \rangle$ of R and transforms $\langle r_{mid}, r_{bou} \rangle$ into privacy-preserving format tr_{mid} and tr_{bou}. The sink sends $\langle T, tr_{mid}, tr_{bou} \rangle$ to storage nodes. When the storage nodes ST receives tr_{mid} and tr_{bou} from the sink, it processes the query through *Distance Comparison Rule* for each data item using cv, tr_{mid} and tr_{bou}. If ST concludes that $d_{i,j}$ is in R according to *Distance Comparison Rule*, it adds $(d_{i,j})_{k_i}$ into the result subset RS_i. Finally, ST merges all RS_i into RS and sends RS to the sink. Upon receiving RS, the sink decrypts the encrypted data in RS and then returns data to users. Fig.3 shows the basic range query processing for 1-dimensional data.

There are three key points in the protocol: construction of characteristic values, transformation from range parameters $\langle r_{mid}, r_{bou} \rangle$ to $\langle tr_{mid}, tr_{bou} \rangle$, and comparison function

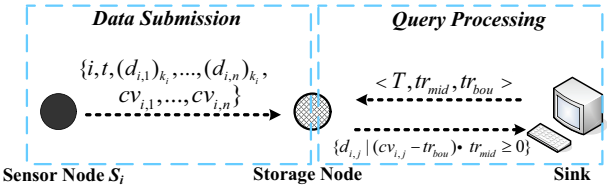


Fig. 3: Basic Idea of secRQ for 1-dimensional Data

for characteristic values $cv_{i,j}$ and $\langle tr_{mid}, tr_{bou} \rangle$. Note that adversary should not infer $d_{i,j}$ from $cv_{i,j}$ without knowing character-transfer function parameters of S_i. Similarly, the adversary also should not infer R from tr_{mid} and tr_{bou} without knowing transform function parameters. Moreover, the comparison result between $cv_{i,j}$ and $\langle tr_{mid}, tr_{bou} \rangle$ should be the correct reflection of relationship between $d_{i,j}$ and R. That is, if $d_{i,j}$ satisfies *Distance Comparison Rule*, the comparison function result should show that $d_{i,j}$ is in R; otherwise, the result should show that $d_{i,j}$ is out of R. Our protocol is inspired by a secure kNN scheme [17]. Next we will describe these key points in details

B. Characteristic Value Construction

First we introduce the construction of characteristic values. During the network initialization phase, the sink first generates a non-singular 3×2 matrix W_s and computes more than N generalized inverse matrices randomly that each generalized inverse matrix M satisfies

$$M \cdot W_s = I_2 \tag{1}$$

where I_x is an identity matrix of size x. Since the number of efficient equations is smaller than that of unknown variables, there exists many M. Obviously, each M is a 2×3 matrix. For example, the sink generates $W_s = \left(\begin{smallmatrix} 1 & 3 & 3 \\ 2 & 2 & 1 \end{smallmatrix}\right)^T$ and M can be $\left(\begin{smallmatrix} 0.25 & -0.75 & 1 \\ 1.5 & -1.5 & 1 \end{smallmatrix}\right)$, $\left(\begin{smallmatrix} 2.5 & -4.5 & 4 \\ 3 & -4 & 3 \end{smallmatrix}\right)$, $\left(\begin{smallmatrix} 1.75 & -3.25 & 3 \\ 2.25 & -2.75 & 2 \end{smallmatrix}\right)$ and so on. Then the sink randomly distributes these generalized inverse matrices to all sensor nodes in the network, and each sensor node is assigned one or more different M, which are represent by M_i. The distribution should guarantee that M_i can only be known by S_i. Since the sink knows all secret keys, one possible method is that the sink allocates $(M_i)_{k_i}$ instead of M_i (each sensor node has a built-in k_i shares with the sink). As a result, routing topology is determined and all sensor nodes obtain their matrices M_i. The sink also owns some generalized inverse matrices of W_s, which are represented by M_s.

Suppose that S_i collects n data items $d_{i,1}, \ldots, d_{i,n}$ at epoch t. S_i first constructs data vector $v_{i,j} = (d_{i,j}, \text{-}0.5(d_{i,j})^2)$ for $d_{i,j}$ and then randomly selects one M_i from its own matrices to construct characteristic value $cv_{i,j}$ through Equation (2).

$$cv_{i,j} = v_{i,j} \cdot M_i \tag{2}$$

$cv_{i,j}$ is a 3-dimensional row vector. For instance, S_i collects two data items 2, 6 at epoch t. S_i first computes $v_{i,1} = (2, \text{-}2)$ and then constructs $cv_{i,1} = (\text{-}1, \text{-}1, 2)$ for data 2 selecting $M_i = \left(\begin{smallmatrix} 2.5 & -4.5 & 4 \\ 3 & -4 & 3 \end{smallmatrix}\right)$. Similarly, S_i constructs $cv_{i,2} = (\text{-}25.5, 22.5, \text{-}12)$ for data 6 selecting $M_i = \left(\begin{smallmatrix} 0.25 & -0.75 & 1 \\ 1.5 & -1.5 & 1 \end{smallmatrix}\right)$. The message that S_i sends to the storage node ST at the end of

epoch t is

$$S_i \rightarrow ST : i, t, \{(d_{i,1}||MV_{i,1})_{k_i}, \ldots, (d_{i,n}||MV_{i,n})_{k_i}\}, \{cv_{i,1}, \ldots, cv_{i,n}\}$$

where $||$ means concatenation and $MV_{i,j}$ is the integrity verification information for $d_{i,j}$ (detailed in Section VI).

C. Range Parameters Transformation

Now we expound the transformation from range parameters $\langle r_{mid}, r_{bou}\rangle$ to $\langle tr_{mid}, tr_{bou}\rangle$ in the sink. When a user proposes the query range $[a, b]$, the sink first constructs two vectors, one is *query middle-value vector* $q_{mid} = u_{mid} \cdot (r_{mid}, 1)$, where u_{mid} is a random positive number, the other is *query bound vector* $q_{bou} = (r_{bou}, \text{-}0.5(r_{bou})^2)$. Then the sink constructs tr_{mid} and tr_{bou} according to Equation (3) and (4).

$$tr_{mid} = W_s \cdot {q_{mid}}^T \tag{3}$$

$$tr_{bou} = q_{bou} \cdot M_s \tag{4}$$

tr_{mid} is a 3-dimensional column vector and tr_{bou} is a 3-dimensional row vector. For instance, a user proposes a query range [4,12], the sink first computes $r_{mid} = 8,\ r_{bou} = 4$, and then transforms $\langle 8, 4\rangle$ into $\langle (20, 52, 50)^T, (\text{-}11, 9, \text{-}4)\rangle$ using $W_s = \left(\begin{smallmatrix} 1 & 3 & 3 \\ 2 & 2 & 1 \end{smallmatrix}\right)^T$, $u_{mid} = 2$ and $M_s = \left(\begin{smallmatrix} 1.75 & \text{-}3.25 & 3 \\ 2.25 & \text{-}2.75 & 2 \end{smallmatrix}\right)$. The message that the sink sends to storage nodes ST is

$$the\ sink \rightarrow ST : T,\ \langle tr_{mid},\ tr_{bou}\rangle$$

where T is the user-specified time segment.

D. Comparison Function

Both the construction of characteristic values and the transformation for range parameters mentioned above are steppingstones to the comparison function for cv, tr_{mid} and tr_{bou} at storage nodes. When the storage node ST receives the range query message, it first picks out the messages received from sensor nodes at epoch t ($t \in T$). ST filters these messages according to **Theorem 2** which guarantees the correctness of comparison.

Theorem 2. *(Range-criterion Rule) Let cv of data item d, tr_{mid} and tr_{bou} of range R be defined as Equation (2), (3), (4) respectively. d is in R if and only if $(cv - tr_{bou}) \cdot tr_{mid} \geqslant 0$.*

Proof: Note that

$$\begin{aligned}
&(cv - tr_{bou}) \cdot tr_{mid} \\
&= (v \cdot M_i - q_{bou} \cdot M_s) \cdot (W_s \cdot {q_{mid}}^T) \\
&= v \cdot M_i \cdot W_s \cdot {q_{mid}}^T - q_{bou} \cdot M_s \cdot W_s \cdot {q_{mid}}^T \\
&= v \cdot (M_i \cdot W_s) \cdot {q_{mid}}^T - q_{bou} \cdot (M_s \cdot W_s) \cdot {q_{mid}}^T \\
&= v \cdot {q_{mid}}^T - q_{bou} \cdot {q_{mid}}^T \\
&= u_{mid} \cdot ((d \cdot r_{mid} - 0.5d^2) - (r_{bou} \cdot r_{mid} - 0.5(r_{bou})^2)) \\
&= 0.5u_{mid}((dis(r_{bou}, r_{mid}))^2 - (dis(d, r_{mid}))^2)
\end{aligned}$$

Since $u_{mid} > 0$, according to **Theorem 1**, the condition is equivalent to

$$(cv - tr_{bou}) \cdot tr_{mid} \geqslant 0 \quad \Leftrightarrow \quad d\ is\ in\ R$$

■

According to **Theorem 2**, we can see our secRQ is a precise range query processing. We call $(cv_{i,j} - tr_{bou}) \cdot tr_{mid}$ the *range-criterion* of $d_{i,j}$. In the previous examples, the *range-criterion* of data 2 is

$$((\text{-}1, \text{-}1, 2) - (\text{-}11, 9, \text{-}4)) \cdot (20, 52, 50)^T = \text{-}20 < 0\ ,$$

and the *range-criterion* of data 6 is

$$((\text{-}25.5, 22.5, \text{-}12) - (\text{-}11, 9, \text{-}4)) \cdot (20, 52, 50)^T = 12 > 0\ ,$$

the storage node filters out the data message including $cv_{i,1} =$ (-1, -1, 2) and keeps the data message including $cv_{i,2} =$ (-25.5, 22.5,-12). For $d_{i,j}$, if its *range-criterion* is non-negative, then ST adds $(d_{i,j}||MV_{i,1})_{k_i}$ into the result subset RS_i. At last, $RS_i = \{i, (d_{i,j}||MV_{i,j})_{k_i} | (cv_{i,j} - tr_{bou}) \cdot tr_{mid} \geqslant 0\}$. After all messages are checked, ST merges all RS_i into RS and returns RS to the sink.

V. Privacy-preserving Range Query Processing for Multi-dimensional Data

In most cases, sensor nodes need to collect multiple kinds of data to monitor different conditions, such as body temperature, blood pressure, heart rate. A z-dimensional data item $D_{i,j}$ collected by sensor node S_i can be denoted as $(d_{i,j}^1, \ldots, d_{i,j}^z)$, where $d_{i,j}^l$ is the l-th dimension value. Similarly, a z-dimensional range query can be denoted as $R_z = ([a^1, b^1], \ldots, [a^z, b^z])$. $D_{i,j}$ is in R_z if and only if $d_{i,j}^l$ is in $[a^l, b^l]$ for each dimension l. We can extend the basic 1-dimensional secRQ protocol in Section IV to z-dimensional secRQ protocol.

Suppose each sensor node S_i collects $D_{i,1}, \ldots, D_{i,n}$ at epoch t, where $D_{i,j} = (d_{i,j}^1, \ldots, d_{i,j}^z)$. Similar to 1-dimensional range query, S_i constructs the *data vector* for each $D_{i,j}$, then builds *data matrix* $V_{i,j}$ for $D_{i,j}$ as

$$V_{i,j} = \begin{pmatrix} d_{i,j}^1 & -0.5(d_{i,j}^1)^2 \\ \vdots & \vdots \\ d_{i,j}^z & -0.5(d_{i,j}^z)^2 \end{pmatrix}$$

Then S_i obtains characteristic value $CV_{i,j}$ of data item $D_{i,j}$ through

$$CV_{i,j} = V_{i,j} \cdot M_i \tag{5}$$

Because M_i is a 2×3 matrix, $CV_{i,j}$ is a $z \times 3$ matrix. The message that S_i sends to the storage node ST is

$$S_i \rightarrow ST : i, t, \{(D_{i,1}||MV_{i,1})_{k_i}, \ldots, (D_{i,n}||MV_{i,n})_{k_i}\}, \{CV_{i,1}, \ldots, CV_{i,n}\}$$

where $MV_{i,j}$ is the integrity verification information for $D_{i,j}$.

When a user proposes a z-dimensional range query R_z, the sink first computes parameters $\langle r_{mid}^l, r_{bou}^l\rangle$ for each dimension of R_z, then builds two matrices Q_{mid} and Q_{bou} through Equation (6) and Equation (7).

$$Q_{mid} = \begin{pmatrix} (u_{mid}^1 \cdot r_{mid}^1) & u_{mid}^1 \\ \vdots & \vdots \\ (u_{mid}^z \cdot r_{mid}^z) & u_{mid}^z \end{pmatrix} \tag{6}$$

$$Q_{bou} = \begin{pmatrix} r_{bou}^1 & -0.5(r_{bou}^1)^2 \\ \vdots & \vdots \\ r_{bou}^z & -0.5(r_{bou}^z)^2 \end{pmatrix} \tag{7}$$

where each u^l_{mid} is a random positive number. Then the sink transforms Q_{min}, Q_{bou} into TR_{mid}, TR_{bou} according to Equation (8) and (9).

$$TR_{mid} = W_s \cdot {Q_{mid}}^T \tag{8}$$

$$TR_{bou} = Q_{bou} \cdot M_s \tag{9}$$

The message that the sink sends to storage node ST is

$$the\ sink \rightarrow ST : T,\ \langle TR_{mid},\ TR_{bou} \rangle$$

While receiving the range query and picking out the data messages satisfying the time segment T, ST filters messages according to **Theorem 3**.

Theorem 3. *Suppose $CV_{i,j}$ of $D_{i,j}$, TR_{mid}, TR_{bou} of range R_z are defined as Equation (5), (8) and (9), respectively. $D_{i,j}$ is in R_z if and only if all the elements on main diagonal of $(CV_{i,j} - TR_{bou}) \cdot TR_{mid}$ are non-negative.*

Obviously, the main diagonal element in the l-th row of $(CV_{i,j} - TR_{bou}) \cdot TR_{mid}$ is the *range-criterion* of the l-th dimension. **Theorem 3** is a dimensional extension of **Theorem 2**. Thus secRQ is also precise under multi-dimensional range queries. After all messages are checked, ST merges all $RS_i = \{i, (D_{i,j}||MV_{i,j})_{k_i} | \text{the main diagonal elements in } (CV_{i,j} - TR_{bou}) \cdot TR_{mid} \text{ are all non-negative}\}$ into RS and returns RS to the sink.

We summarize the whole privacy-preserving range query processing in **Protocol 1**.

Protocol 1. secRQ for z-dimensional Data

- Network initialization phase: The sink secretly generates a non-singular 3×2 matrix W_s, and allocates 2×3 matrices M_i satisfying Equation (1) to each sensor nodes.
- Data-sensing phase: Suppose S_i collects $D_{i,1}, \ldots, D_{i,n}$ at epoch t. For each data item $D_{i,j}$, S_i generates integrity verification information $MV_{i,j}$ (detailed in Section VI) and then computes $CV_{i,j}$ according to Equation (5). After encryption, S_i sends data message $\{i, t, \{(D_{i,1}||MV_{i,1})_{k_i}, \ldots, (D_{i,n}||MV_{i,n})_{k_i}\}, \{CV_{i,1}, \ldots, CV_{i,n}\}$ to its storage node ST.
- Query-proposing phase: For a range query $\langle T, R_z \rangle$, the sink computes parameters $\langle r^l_{mid}, r^l_{bou} \rangle$ for the l-th dimension range and transforms these parameters to $\langle TR_{mid}, TR_{bou} \rangle$ according to Equation (8) and (9). Then it sends $\langle T, TR_{mid}, TR_{bou} \rangle$ to ST.
- Query-processing phase: Receiving $\langle T, TR_{mid}, TR_{bou} \rangle$, ST first picks out the messages satisfying $t \in T$, then computes *range-criterions* and filters messages according to **Theorem 3**. Finally, ST merges all $RS_i = \{i, (D_{i,j}||MV_{i,j})_{k_i} | \text{the main diagonal elements in } (CV_{i,j} - TR_{bou}) \cdot TR_{mid} \text{ are all non-negative}\}$ into RS and sends RS to the sink.

VI. Data Integrity Verification for Range Query Processing

In a malicious environment, storage nodes may be compromised to reply damaged (fake or incomplete) results to the sink, which makes users receive the damaged result and influences the correctness of users' decision. As a result, it is necessary to verify data integrity of results. Data integrity generally refers to two aspects. One is that the final result should not include forged data items, the other is that the final result should not exclude any data item satisfying the query. To check the integrity, we present a novel verification scheme called *mutual verification* via constructing specific relationships between any two data items. In the following, we first give some definitions used in our integrity verification scheme, and then elaborate the *mutual verification* scheme based upon these definitions. To ease representation, sensor node ID i is omitted in some place.

A. Definitions

For clear description, we define three important definitions: *combined space*, *pure relationship* and *mutual verification object*. We will use these definitions to verify integrity in the *mutual verification* scheme.

1) ***Combined Space****:* Considering two z-dimensional data items denoted as $D_\alpha(d^1_\alpha, \ldots, d^z_\alpha)$ and $D_\beta(d^1_\beta, \ldots, d^z_\beta)$, we can establish a *combined space* $CS_{\alpha,\beta} = ([d^1_\alpha, d^1_\beta], \ldots, [d^z_\alpha, d^z_\beta])$. If $d^l_\alpha > d^l_\beta$, the sensor node changes $[d^l_\alpha, d^l_\beta]$ to $[d^l_\beta, d^l_\alpha]$. For example, there are five 2-dimension data items $D_1(5,4)$, $D_2(9,2)$, $D_3(13,11)$, $D_4(1,7)$, $D_5(9,8)$ in Fig.4. We observe that D_1 and D_2 can form *combined space* $CS_{1,2} = ([5,9][2,4])$. Similarly, $CS_{1,3} = ([5,13][4,11])$, $CS_{1,4} = ([1,5][4,7])$, $CS_{1,5} = ([5,9][4,8])$.

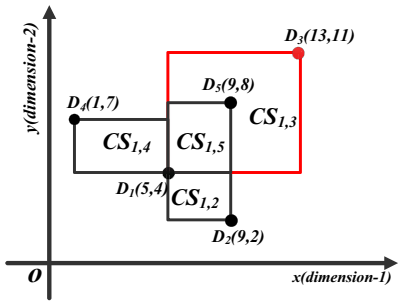


Fig. 4: Mutual Verification

2) ***Pure Relationship****:* Construct $CS_{\alpha,\beta}$ for D_α and D_β according to the definition of *combined space*. If $CS_{\alpha,\beta}$ does not contain any other data item, we call that D_α and D_β have *pure relationship*. In Fig.4, D_1 has *pure relationship* with D_2, D_4 and D_5, respectively. Obviously, D_1 does not have *pure relationship* with D_3, because the *combined space* $CS_{1,3} = ([5,13][4,11])$ includes another data item $D_5(9,8)$.

3) ***Mutual Verification Object****:* The *mutual verification object* MV_j of the data item D_j is defined as the data set in which each data item has *pure relationship* with D_j. For example, in Fig.4, $MV_1 = (D_2||D_4||D_5)$, $MV_2 = (D_1||D_5)$, $MV_3 = (D_5)$, $MV_4 = (D_1||D_5)$, $MV_5 = (D_1||D_2||D_3||D_4||D_5)$.

B. Mutual Verification Scheme

Now we describe how to use *mutual verification* scheme to verify the integrity of query result in sensor nodes, storage nodes and the sink respectively.

1) Sensor nodes: Sensor nodes are responsible for generating and embedding *mutual verification objects* into the original data items. S computes MV_j for each data item D_j collected at each epoch t. Then S sends $(D_j||MV_j)_k$ to its storage node ST. Fig.5 shows that the sensor node S collects data items $D_1(5,4), D_2(9,2), D_3(13,11), D_4(1,7), D_5(9,8)$ at epoch t. Then S computes corresponding $MV_1 = (D_2||D_4||D_5)$, $MV_2 = (D_1||D_5)$, $MV_3 = (D_5)$, $MV_4 = (D_1||D_5)$, $MV_5 = (D_1||D_2||D_3||D_4||D_5)$ as shown in Fig.4. S encrypts each data item D_j and MV_j together, and submits $(D_1||MV_1)_k$, $(D_2||MV_2)_k$, $(D_3||MV_3)_k$, $(D_4||MV_4)_k$, $(D_5||MV_5)_k$ to the storage node ST.

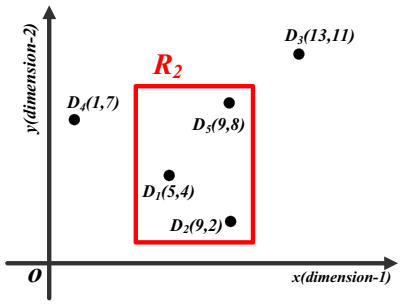


Fig. 5: 2-dimensional Range Query

2) Storage node: ST processes range query $R_z = ([a^1, b^1], \ldots, [a^z, b^z])$ and returns RS to the sink according to **Protocol 1**. In Fig.5, given $R_2 = ([4,10],[1,9])$, then $(D_1||MV_1)_k, (D_2||MV_2)_k, (D_5||MV_5)_k$ should be added in RS.

3) The sink: Upon receiving RS, the sink decrypts RS to restore DS and VS, where DS denotes the plaintext data set containing D_j, and VS denotes the mutual verification object set containing MV_j. In Fig.5, $DS = \{D_1, D_2, D_5\}$, $VS = \{MV_1, MV_2, MV_5\}$. The sink verifies the query result integrity following three principles:

(a) According to DS and VS, get the original data distribution. Since secRQ is a precise range query protocol, if any data item in DS does not satisfy the query range R_z, the query result is damaged.

(b) Every D_j in DS and its corresponding MV_j in VS should satisfy *pure relationship*. If not, the query result is damaged.

(c) Let DS' denote the data set satisfying the query range R_z in VS. For a sensor node S, there are three cases should be considered. (1) $DS = \varnothing$. There should be $DS' = \varnothing$. (2) Only one data item in DS. There should be $DS' = \varnothing$. (3) More data items than one in DS. If DS is complete, DS should be equal to DS'. In Fig.4, $D_1 \in MV_2$ and MV_5, $D_2 \in MV_1$ and MV_5, $D_5 \in MV_1$, thus $DS = DS' = \{D_1, D_2, D_5\}$. If $(D_1||MV_1)_k$ is deleted in storage node ST, the sink can detect the loss of D_1 through $DS = \{D_2, D_5\} \neq DS =' \{D_1, D_2, D_5\}$, where DS' is generated by $VS = \{MV_2, MV_5\}$.

VII. Analysis

A. Complexity Analysis

We mainly analyze computation complexity. Suppose there are N sensor nodes, each sensor node collects n z-dimensional data items at epoch t. We divide the range query processing into four phases as shown in **Protocol 1** to analyze complexity.

- Network initialization phase: The sink allocates M to each sensor node, the complexity is $O(N)$.
- Data-sensing phase: The complexity of computing characteristic value is $O(1)$, thus for n z-dimensional data items, the total complexity is $O(zn)$. Similarly, the complexity of integrity verification construction is $O(zn)$.
- Query-proposing phase: For a z-dimensional range query, the sink first computes $\langle r^l_{mid}, r^l_{bou} \rangle$ for each dimension. The transformation for each dimension is $O(1)$, thus the complexity of this phase is $O(z)$.
- Query-processing phase: The storage node ST checks each characteristic value received from sensor nodes, and the complexity of this phase is $O(zn)$.

B. Privacy Analysis

We analyze the privacy-preserving ability of secRQ in the following aspects.

- Sensor node is compromised. Since the adversary only knows the data items of the compromised sensor node, this case has little influence on network security. Even if the adversary obtains characteristic value construction, it does not know the characteristic values of other sensor nodes' sensitive data as well as the transformed queries. This case does not threaten the security of the network.
- Storage node is compromised. The adversary can obtain $(D_{i,j})_{k_i}$, $CV_{i,j}$ and $\langle TR_{mid}, TR_{bou} \rangle$. Since the adversary does not know k_i, it cannot decrypt $(D_{i,j})_{k_i}$ to get $D_{i,j}$. Without knowing characteristic value construction, the adversary cannot recover data item $D_{i,j}$ from $CV_{i,j}$. Similarly, Q_{mid} and Q_{bou} are unrecoverable due to the lack of range parameters transformation. Therefore a single compromised storage node has little effect on data privacy and query privacy in our scheme. A compromised storage node may return fake and incomplete results, we will demonstrate that our integrity verification can detect fake results in next subsection.
- Collusion attack. Since the sink is credible in our system model, the adversary does not know the generation rule of M as shown in Equation (1). Two cases should be considered under collusion attacks. One case is that the adversary compromises some sensor nodes. The adversary does not know characteristic values of other sensor nodes' sensitive data, so this case is similar to the first aspect and has little influence on network security. The other case is that the adversary compromises both sensor nodes and their storage node. Although the adversary obtains

characteristic value construction from compromised sensor nodes, each sensor nodes holds different M, and the adversary cannot deduce M of other sensor nodes from these compromised sensor nodes without knowing generation rule of M. Without knowing M of other sensor nodes, the adversary cannot know the sensitive data items from their characteristic values. Although the adversary knows the comparison function $(CV_{i,j} - TR_{bou}) \cdot TR_{mid}$, it cannot deduce the range parameters transformation from this comparison function, therefore Q_{mid} and Q_{bou} are still unrecoverable.

- Probability attack. Our protocol is under many-to-many model, that is, each data item has some different one characteristic values, and different data items may generate the same characteristic value, so do the query range and transformed query matrices. The sink and each sensor node may have one or more M, and they can randomly choose different M to compute TR_{bou} and CV. Thus same r_{bou} or same data items may have different TR_{bou} or CV. Similarly, to resist probability attacks, the sink can build different TR_{mid} via using different u^l_{mid}.

C. Integrity Analysis

In *mutual verification* scheme, the sink can correctly verify the integrity of query results. One data item D must exist in *mutual verification object* of other data items which have *pure relationships* with D. Meanwhile, the *mutual verification* scheme has strong scalability, which is not only adaptable to 1-dimensional data, but also highly suitable for multi-dimensional data. For 1-dimensional data, the *mutual verification* scheme is analogous to neighborhood chains [4]. For multi-dimensional data, due to the definition of *combined space*, our scheme can be extended to multiple dimensions. There is a mutual verification object for each multi-dimensional data item for checking integrity.

In this paper, two types of integrity attacks are taken into consideration. First, fake data items may be inserted into RS. In our scheme, because MV_j is attached to data item D_j and encrypted before submission, it is impossible for the compromised storage node to falsify D_j with correct MV_j without knowing the secret key. Second, some data items may be deleted from RS. Next, we give the theoretical analysis and proof to confirm the *mutual verification* scheme can detect this illegal deletion. Only when the query result strictly abides by principles (a), (b), (c) in Section VI.B, can the sink assure the integrity of result. Principle (a) and (b) allow the sink to check whether each decrypted data item in RS satisfies the query range R. Principle (c) enables the sink to verify if some data items satisfying the query range are removed illegally. Principles (a), (b) and the first two cases in Principle (c) are facile to understand. We mainly prove the third case in Principle (c) with **Theorem 4** and **Theorem 5**, *i.e.*, both **Theorem 4** and **Theorem 5** are under the condition that more data items than one are in DS.

Theorem 4. *A data item D_i $(D_i \in DS)$ is **isolated** if $\forall D_j \in DS, j \neq i, D_i \notin MV_j$. When there are more data items than one in DS, any $D_i \in DS$ can not be isolated.*

Proof: We use *Reduction ad absurdum* to prove the theorem. Assume that $\exists\ D_i,\ \forall D_j \in DS, j \neq i, D_i \notin MV_j$. If $D_i \notin MV_j$, due to the definition of *Pure Relationship*, we know that $\exists D_\eta, \eta \neq i, j, s.t.\ D_\eta$ falls into $CS_{i,j}$. Considering $D_i, D_j \in DS$ and the properties of *combined space*, $CS_{i,j}$ should be covered by R. By reason of $D_\eta \in CS_{i,j} \subset R$, we can know that $D_\eta \in DS$.

(i) If $CS_{i,\eta} \cap DS = \{D_i, D_\eta\}$, then $D_i \in MV_\eta$. It is contradictory to the assumption.

(ii) If $CS_{i,\eta} \cap DS \neq \{D_i, D_\eta\}$, then $D_i \notin MV_\eta$. So $\exists D_{\eta_1}, \eta_1 \neq i, j, \eta, s.t.\ D_{\eta_1} \in CS_{i,\eta}$ and $D_{\eta_1} \in DS$. If $CS_{i,\eta_1} \cap DS = \{D_i, D_{\eta_1}\}$, then $D_i \in MV_{\eta_1}$. It is similar with (i). In contrast, $D_i \notin MV_{\eta_1}$ and $\exists D_{\eta_2}, \eta_2 \neq i, j, \eta, \eta_1, s.t.\ D_{\eta_2} \in CS_{i,\eta_1}$ and $D_{\eta_2} \in DS$. It is also similar to the condition before. On the analogy of this, $D_{\eta_{n-1}}, \eta_{n-1} \neq i, j, \eta, \cdots, \eta_{n-2}, s.t.\ D_{\eta_{n-1}} \in CS_{i,\eta_{n-2}}$ and $D_{\eta_{n-1}} \in DS$.

(iii) Owing to the total order [18] of *combined space*, we obtain that $CS_{i,\eta_{n-1}} \subset CS_{i,\eta_{n-2}} \subset \cdots \subset CS_{i,\eta}$. Since $\lim\limits_{n\to\infty} sup_{\xi,\zeta \in CS_{i,\eta_{n-1}}} \rho(\xi, \zeta) = 0$, on the basis of the Closed Nested Interval Theorem [19] , there is an unique data item $D_{\eta_n}, s.t.\ D_{\eta_n} \in CS_{i,\eta_{n-1}}, D_{\eta_n} \in DS$ and $CS_{i,\eta_n - 1} \cap DS = \{D_i, D_{\eta_n}\}$. It is similar with (i) contradictory to the assumption.

Thus, if there are more data items than one in DS and $D_i \in DS$, D_i can not be *isolated*. ■

Theorem 5. *If DS is complete, $DS = DS'$.*

Proof: $DS = DS' \Leftrightarrow DS \subset DS'$ and $DS' \subset DS$. When DS is complete, considering the definition of DS and DS', we know that $DS' \subset DS$. Next we should prove $DS \subset DS'$. $DS \subset DS' \Leftrightarrow \forall D_i \in DS, \exists D_j \in DS, j \neq i, D_i \in MV_j$. According to **Theorem 4**, we know that any D_i $(D_i \in DS)$ can not be *isolated*, *i.e.*, $\forall D_i \in DS, \exists D_j \in DS, j \neq i, D_i \in MV_j$. Thus we have $DS \subset DS'$. Therefore, if DS is complete, $DS = DS'$. ■

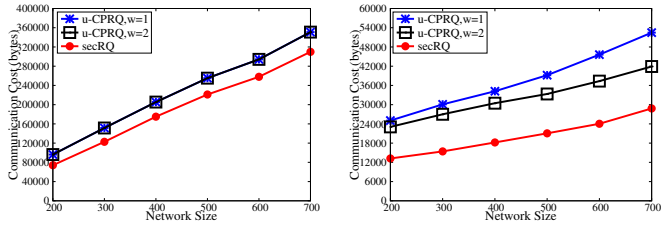
Now we discuss the detection probability of the misbehavior that a storage node removes data items from the query result. Because the original data items have been encrypted and each *mutual verification object* MV_j of data item D_j has been attached, the only option left for the adversary is deleting all the data items and their corresponding *mutual verification objects*. Let A_i denote the event that data item D_i in the query result is removed. The detection probability is not 100% if and only if every data item D_i and its corresponding MV_i in RS are all deleted. Therefore, the detection probability can be formulated as:

$$1 - \prod_i \prod_j P(A_j | A_i,\ D_i, D_j \in Q, D_j \in MV_i, i \neq j)$$

The above analysis demonstrates that the *mutual verification* scheme can detect fake and incomplete results with high probability and strong scalability.

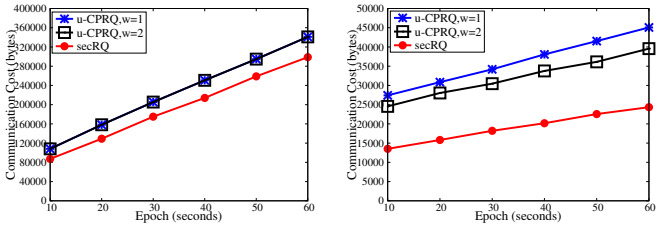
VIII. Performance evaluation

In this section, we evaluate secure range query processing secRQ in terms of communication cost and accuracy. We compare our secRQ with u-CPRQ [10], which can also



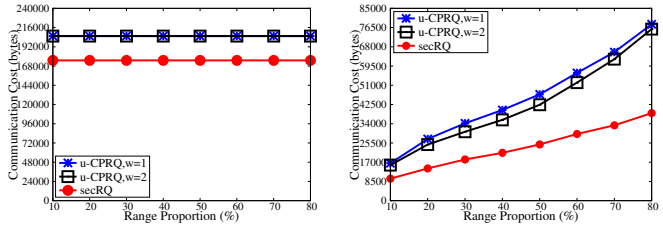
(a) Sensor Nodes (b) Storage Nodes

Fig. 6: Impact of Network Size on Communication Cost



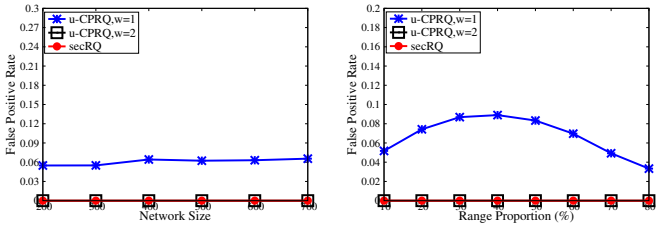
(a) Sensor Nodes (b) Storage Nodes

Fig. 7: Impact of Epoch on Communication Cost



(a) Sensor Nodes (b) Storage Nodes

Fig. 8: Impact of Range Proportion on Communication Cost



(a) Network Size (b) Range Proportion

Fig. 9: False Positive Rate

resist collusion attacks. We implement these protocols on OMNet++4.1 [20], and our 2-dimensional data set comes from Grand-St-Bernard Deployment [21], a small SensorScope network which deployed at the Great St. Bernard Pass between Switzerland and Italy to monitor meteorology.

The area of the network is set to $400m \times 400m$. We use the number of sensor nodes to represent network size. Sensor nodes are uniformly deployed in the network. Assume the network is separated into 4 identical regions and a storage node is placed at the center of each region. The transmission radius of each sensor node is $50m$. Without loss of generality, we use 2-dimensional data items and each domain is $[0, 100]$. Thus the domain length of each dimension is 100. We use 128-bit Data Encryption Standard (DES) as the data encryption. In addition, for u-CPRQ, we use conditions as in [10], *i.e.* the Bloom filter is 128-bit, the hash function number of the sink is 4 and at least 3 hash functions are distributed to each sensor node. To reduce false positives, u-CPRQ is improved by setting $w = 2$ where w means the number of Bloom filter used for range query. Although this improvement does not increase communication cost of both sensor nodes and storage nodes, it introduces more computation cost to storage nodes.

Our experiment is under *format-transmission*, which means except sensor node IDs and epoch t (both are 8-bit), other data items should transmitted in either integer format (32-bit) or float format (32-bit). Unless otherwise stated, the network size N=400, epoch t=30s, range proportion (the proportion of $|b - a|$ to each domain length) is 40%. We generated 200 random range queries for each turn.

A. Communication

In WSNs, communication is the dominant factor to consume energy which affects network lifetime. Since the sink is resource-rich, the communication cost produced by the sink is not be considered. Communication cost is usually measured by the number of bits or bytes transmitted from sensor nodes to storage nodes and that transmitted from storage nodes to the sink, *i.e.* communication cost of sensor nodes and that of storage nodes.

Fig.6 and Fig.7 show the impact of network size and epoch on communication cost respectively. It can be observed that the communication cost of these protocols increases with network size and epoch, and secRQ consumes less communication than u-CPRQ. The reason is given as follows. Larger network size means more data items are collected by sensor nodes and transmitted to storage nodes, and larger epoch means sensor nodes upload more data items to storage node in each submission. Under *format-transmission*, u-CPRQ should use at least three numbers to represent Bloom filter code of each data and extra two Bloom filter codes as bounds of the message, and w used in u-CPRQ does not affect the communication cost of sensor nodes. By contrast, secRQ only uses a 3-dimensional vector to represent each data item. Therefore secRQ consumes less communication cost of sensor nodes. In u-CPRQ, storage nodes return each eligible encrypted data item as well as their Bloom filter codes to the sink to verify data integrity. When $w = 1$, u-CPRQ also produces false positives which further increase communication cost of storage nodes. In secRQ, the storage nodes do not return characteristic values since the integrity verification information is embedded into the encrypted data items, thus secRQ reduces the communication cost for storage nodes.

Fig.8 shows the impact of range proportion on communication cost. Given the fixed network size and epoch, range proportion does not affect communication cost of sensor nodes. It can be seen that our secRQ consumes less communication cost than u-CPRQ. the reason is similar to that in Fig.6 and Fig.7. Larger range proportion means more data items satisfy the range query, therefore storage nodes return a larger result set RS to the sink. For storage nodes, besides smaller result set structure, secRQ is precise while u-CPRQ has false positives. It is related to the result accuracy, which is discussed in next subsection. Therefore secRQ consumes less communication cost of storage nodes under different range proportions.

B. Accuracy

In our experiment, we define *false positive rate* as the ratio of the number of unsatisfactory data items in the result to the number of data items in the result. Obviously, higher *false positive rate* means lower accuracy.

Fig.9 shows the impact of network size and range proportion on *false positive rate*. Our experiment results show that the impact of epoch on *false positive rate* is similar to the impact of network size, therefore we just exhibit the impact of network size. From Fig.9 we can see that secRQ has no false positive as well as u-CPRQ when $w = 2$. Fig.9(a) shows that *false positive rates* of these protocols are stable under different network sizes. Fig.9(b) exhibits the impact of range proportion on *false positive rate*. According to **Theorem 2** and **Theorem 3**, secRQ is a precise range query protocol which has no false positive. However, range proportion concerns the number of data items included in Bloom filter codes of queries. u-CPRQ may have false positives due to the property of Bloom filter [22]. When $w = 2$, u-CPRQ can also be a precise range query protocol [10]. Evaluation results in Fig.9 also confirm the accuracy of secRQ.

IX. Conclusion

In this paper, we present a secure and efficient range query protocol secRQ that can resist collusion attacks and probability attacks in tiered WSNs. secRQ preserves data privacy, enables storage nodes to process range queries precisely and supports the sink to defend against compromised storage nodes even if the network faces collusion attacks and probability attacks. Besides, we propose *mutual verification* scheme to verify the validity of query results. Thorough analysis and simulation results confirm the high performance of secRQ in terms of privacy preservation, integrity detection, efficiency and accuracy.

Acknowledgment

This work is supported by the National High Technology Research and Development Program of China (863 Program) (No. 2014AA015204), the National Basic Research Program of China (973 Program) (No. 2014CB340402, 2012CB316205), the National Natural Science Foundation of China (Grant No. 61070056, 61272137, 61202114) and National Social Science Funds of China (No. 12&ZD220). This work was partially done when the authors visited SA Center for Big Data Research hosted at Renmin University of China. This Center is partially funded by the Chinese National "111" Project "Attracting International Talents in Data Engineering and Knowledge Engineering Research".

References

[1] O. Gnawali, K.-Y. Jang, J. Paek, M. Vieira, R. Govindan, B. Greenstein, A. Joki, D. Estrin, and E. Kohler, "The tenet architecture for tiered sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*. ACM, 2006, pp. 153–166.

[2] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-centric storage in sensornets with ght, a geographic hash table," *Mobile networks and applications*, vol. 8, no. 4, pp. 427–442, 2003.

[3] Y. Yao, N. Xiong, J. H. Park, L. Ma, and J. Liu, "Privacy-preserving max/min query in two-tiered wireless sensor networks," *Computers & Mathematics with Applications*, vol. 65, no. 9, pp. 1318–1325, 2013.

[4] F. Chen and A. X. Liu, "Safeq: Secure and efficient query processing in sensor networks," in *IEEE International Conference on Computer Communications*, 2010, pp. 2642–2650.

[5] X. Zhang, L. Dong, H. Peng, H. Chen, D. Li, and C. Li, "Achieving efficient and secure range query in two-tiered wireless sensor networks," in *IEEE/ACM International Symposium on Quality of Service*, 2014.

[6] B. Sheng and Q. Li, "Verifiable privacy-preserving range query in two-tiered sensor networks," in *IEEE International Conference on Computer Communications*, 2008, pp. 46–50.

[7] F. Chen and A. X. Liu, "Privacy and integrity-preserving range queries in sensor networks." in *IEEE/ACM Transactions on Networking*, vol. 20, 2012, pp. 1774–1787.

[8] G. Li, L. Guo, X. Gao, and M. Liao, "Bloom filter based processing algorithms for the multi-dimensional event query in wireless sensor networks," *Journal of Network and Computer Applications*, vol. 37, pp. 323–333, 2014.

[9] Y. Yi, R. Li, F. Chen, A. X. Liu, and Y. Lin, "A digital watermarking approach to secure and precise range query processing in sensor networks," in *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013, pp. 1950–1958.

[10] X. Zhang, L. Dong, H. Peng, H. Chen, S. Zhao, and C. Li, "Collusion-aware privacy-preserving range query in tiered wireless sensor networks," *Sensors*, vol. 14, no. 12, pp. 23 905–23 932, 2014.

[11] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: Privacy-preserving data aggregation in wireless sensor networks," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. IEEE, 2007, pp. 2045–2053.

[12] H. Liu, H. Wang, Y. Chen, and D. Jia, "Defending against frequency-based attacks on distributed data storage in wireless networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 3, pp. 209–245, 2014.

[13] J. Cheng, H. Yang, S. H. Wong, P. Zerfos, and S. Lu, "Design and implementation of cross-domain cooperative firewall," in *Network Protocols, 2007. ICNP 2007. IEEE International Conference on*. IEEE, 2007, pp. 284–293.

[14] A. X. Liu and F. Chen, "Collaborative enforcement of firewall policies in virtual private networks," in *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing*. ACM, 2008, pp. 95–104.

[15] F. Chen and A. X. Liu, "Privacy and integrity preserving multi-dimensional range queries for cloud computing," in *Networking Conference, 2014 IFIP*. IEEE, 2014, pp. 1–9.

[16] R. Szewczyk and A. Ferencz, "Energy implications of network sensor designs," *Berkeley Wireless Research Center, Santa Clara, Calif, USA*, 2000.

[17] W. K. Wong, D. W.-l. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*. ACM, 2009, pp. 139–152.

[18] R. Nederpelt and F. Kamareddine, *Logical reasoning: a first course, Chapter 20.2: Ordered Sets. Orderings*. King's College, 2004.

[19] J. A. Fridy, *Introductory analysis: The theory of calculus, Chapter 3.3 The Nested Intervals Theorem*. Gulf Professional Publishing, 2000.

[20] "Omnet++4.1," http://www.omnetpp.org.

[21] "Grand-st-bernard deployment," http://lcav.epfl.ch/cms/lang/en/pid/86035.

[22] "Bloom filter," http://en.wikipedia.org/wiki/Bloom_filter.