# Secure Multidimensional Range Queries in Sensor Networks

Rui Zhang, Jing Shi, and Yanchao Zhang
Electrical and Computer Engineering Department
New Jersey Institute of Technology
Newark, NJ 07102-1982, USA
{rz23,js39,yczhang}@njit.edu

## ABSTRACT

Most future large-scale sensor networks are expected to follow a two-tier architecture which consists of resource-rich master nodes at the upper tier and resource-poor sensor nodes at the lower tier. Sensor nodes submit data to nearby master nodes which then answer the queries from the network owner on behalf of sensor nodes. Relying on master nodes for data storage and query processing raises severe concerns about data confidentiality and query-result correctness when the sensor network is deployed in hostile environments. In particular, a compromised master node may leak hosted sensitive data to the adversary; it may also return juggled or incomplete query results to the network owner. This paper, for the first time in the literature, presents a suite of novel schemes to secure multidimensional range queries in tiered sensor networks. The proposed schemes can ensure data confidentiality against master nodes and also enable the network owner to verify with very high probability the authenticity and completeness of any query result by inspecting the spatial and temporal relationships among the returned data. Detailed performance evaluations confirm the high efficacy and efficiency of the proposed schemes.

## Categories and Subject Descriptors

C.2.0 [**Computer Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Wireless communication*

## General Terms

Design, Performance, Security

## Keywords

Sensor networks, multidimensional range query, security

## 1. INTRODUCTION

Many sensor networks are expected to be deployed in remote and extreme environments such as oceans, volcanos, animal habitats, and battlefields. It is often impossible or prohibitive to maintain a stable always-on communication connection from the sensor network to the external network owner. This situation necessitates in-network data storage [1–4] such that data continuously produced by sensor nodes are stored inside the network. The network owner can access the data when needed via an ad-hoc communication connection (e.g., a satellite link) or by physical means like dispatching robots to the sensor network [2].

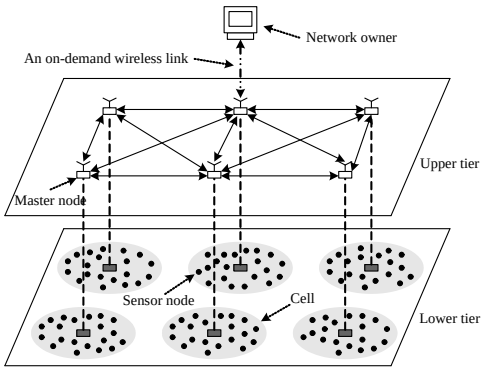



**Figure 1: An abstract two-tier sensor network architecture.**

Despite rapid progress in storage technology, it remains economically infeasible to furnish individual sensor nodes, which may be in tens of thousands, with large storage space. It is, however, viable to equip relatively fewer nodes with several gigabytes of NAND flash storage for a few tens of dollars [2]. Consider Fig. 1 as an example. Such special nodes, which we call *master node*s, collect data from nearby sensor nodes, store them locally for extended periods of time, and answer various ad-hoc data queries from the network owner. Master nodes can also have abundant resources in energy and computation and form a multi-hop wireless mesh network among themselves using long-range high-bandwidth radios. Such a two-tier network architecture is known to be indispensable for increasing network capacity and scalability, reducing system management complexity, and prolonging network lifetime [1, 5].

Relying on master nodes for data storage and query processing raises significant security concerns. For instance, if a sensor network is deployed in hostile military or homeland security scenarios, master nodes are attractive targets of attack and may be compromised by the adversary. As another example, we envision that in the near future there might be sensor networks built to provide commercial data service which may face various attacks from business competitors. The adversary may launch two types of severe attacks through compromised master nodes. First, the adversary can read all the data stored on them which are possibly very sensitive (e.g., intrusion events). This attack calls for sound defenses to ensure data confidentiality while still enabling efficient data query

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
*MobiHoc'09,* May 18–21, 2009, New Orleans, Louisiana, USA.
Copyright 2009 ACM 978-1-60558-531-4/09/05 ...$5.00.

processing. Second, the adversary may instruct compromised master nodes to return juggled and/or incomplete data in response to ad-hoc queries from the network owner. This attack is more subtle and harmful than blind DoS attacks on the sensor network, especially when the query results are used as the basis for making critical military or business decisions. To defend this attack, we must enable the network owner to check the *authenticity* and *completeness* of any query result. The term authenticity means that all the data in the result originated from the purported sources and have not been tampered with, and completeness means that the result includes all the data satisfying the query.

Range queries are an important and common type of queries in sensor networks which ask for data with one or multiple attributes falling in specified ranges (called *one-dimensional* or *multidimensional* range queries) [6, 7]. An exemplary multidimensional range query is "Return all observed objects with weights between 170 and 220 pounds and moving speeds between 3 and 5 miles per hour." This paper is concerned with supporting secure range queries, especially multidimensional range queries, against possibly compromised master nodes. Extending our work to other types of data queries [8] is left for future work.

To the best of our knowledge, secure range queries in sensor networks has received attention only recently [4, 9]. Aiming at one-dimensional range queries, both schemes [4, 9] could ensure data confidentiality and also enable query-result authenticity and completeness verification with different communication and computation overhead. How to secure more general multidimensional range queries, however, remains an open challenging task.

This paper, for the first time in the literature, investigates techniques to secure multidimensional range queries in two-tier sensor networks. We employ the bucketing technique [10, 11] to achieve data confidentiality and also query-result authenticity verification while ensuring efficient query processing (§ 3). Our major contributions are a suite of novel techniques for the network owner to verify query-result completeness (§ 4). In particular, our first construction is a deterministic approach that is an extension of the technique in [4] to multidimensional cases (§ 4.1). It allows the network owner to immediately catch misbehaving master nodes at the cost of communication overhead growing exponentially with the number of dimensions or queriable data attributes. We then present two novel probabilistic techniques with significantly less communication overhead, including a spatial crosscheck technique and a temporal crosscheck technique. The former aims to create some relationships among data generated by sensor nodes affiliated with the same master node (§ 4.2), while the latter aims to embed some relationships among data produced in different time periods (§ 4.3). These two techniques can collectively allow the network owner to verify with overwhelming probability whether a query result is complete by examining the spatial and temporal relationships among the returned data. We further propose a random probing technique as a complement to spatial and temporal crosscheck techniques to cope with compromised sensor nodes (§ 4.4). Built upon symmetric cryptographic primitives, our techniques are shown to be very effective and efficient through comprehensive theoretical analysis and performance evaluations.

# 2. NETWORK, QUERY, AND ADVERSARY MODELS

## 2.1 Network Model

We consider a large-scale two-tier sensor network with thousands of resource-poor sensor nodes and relatively fewer resource-rich master nodes, as shown in Fig. 1. Master nodes have abundant resources in storage, energy (a solar panel and/or heavy-duty rechargeable batteries), and computation; they also communicate in a multi-hop fashion via relatively long-range high-rate radios. In contrast, sensor nodes are more constrained in storage, energy, computation, and communication capabilities.

Each master node is in charge of a physical region of the network field, called a *cell*. Sensor nodes in a cell are affiliated with the master node in that cell. Here we follow the conventional assumption that master nodes and sensor nodes know their respective geographic locations and also which cell they are in, which can be realized by many existing techniques such as [12, 13]. Depending on concrete applications, the cells of two neighboring master nodes may overlap, in which case sensor nodes in the overlapping region are affiliated with both master nodes.

We do not assume an always-on communication connection to the external network owner. Instead, the network owner can query data by an on-demand wireless link (e.g., a satellite link) connected to some master node(s). To prevent storage overflow of master nodes, mobile sinks [14] can also be periodically (e.g., quarterly) dispatched to collect data and empty the storage of master nodes.

As in [4, 9], we assume that time is divided into *epoches* and that sensor and master nodes are loosely synchronized. At the end of each epoch, each sensor node submits to its master node all the data (if any) it produced during that epoch. Without loss of generality, we subsequently focus on a cell $C$ with $N$ sensor nodes $\{S_i\}_{i=1}^{N}$ and a compromised (yet undetected) master node $\mathcal{M}$. It is worth noting that all the operations also apply to all the other cells with or without compromised master nodes.

## 2.2 Multidimensional Range Queries

Event data generated by sensor nodes can generally be described as a tuple of attribute values $\{A_j\}_{j=1}^{\mathbf{d}}$, where $\mathbf{d} \geq 1$ depends on concrete sensor network applications. Each attribute $A_j$ represents a sensor reading or an aspect of the event such as the weight of an observed object, its location, its speed and moving trajectory, or its appearance or lingering time. Let $\mathcal{A} \subseteq \{A_j\}_{j=1}^{\mathbf{d}}$ be a subset of attributes the network owner is interested in. For sake of simplicity, we will focus on the following type of *primitive* multidimensional range queries,

$$(\text{cell } = C) \wedge (\text{epoch } = t) \bigwedge_{A_a \in \mathcal{A}} (l_a \leq A_a \leq h_a), \quad (1)$$

where $C$ and $t$ denote the cell ID and the interested epoch, respectively, and $[l_a, h_a]$ is the interested range of attribute $A_a$. For other types of range queries which, for example, involve multiple epoches and/or cells or the union of attributes, they can be converted into multiple primitive range queries. Our work can also be easily extended to support range queries concerning specific sensor nodes. Note that prior work [4, 9] aims at single-attribute (or one-dimensional) range queries, which are a special case of ours with only one queriable attribute (i.e., $\mathbf{d} = 1$).

## 2.3 Adversary Model

Tremendous efforts have been made to secure sensor network communications, see for example [15–22]. This paper focuses on secure multidimensional range queries, an open challenge. We resort to the existing rich literature for other important issues such as key management, secure routing, broadcast authentication, secure localization, DoS mitigation, and particularly secure and reliable message transmissions.

Although the adversary may directly compromise sensor nodes to read their data and manipulate their behavior, it is much more tempting to take over master nodes for their significant roles in the

two-tier sensor network. The adversary is assumed to have compromised some master nodes whereby to launch attacks against data confidentiality and query-result authenticity and completeness. The adversary may also compromise sensor nodes to aid compromised master nodes. We, however, follow the conventional assumption that non-compromised sensor nodes are always the majority. Since every master node is only responsible for its own cell, the collusion of compromised master nodes will not do more harm. Our subsequent discussion thus concentrates on one compromised master node $\mathcal{M}$ in charge of a cell with $N$ sensor nodes $\{S_i\}_{i=1}^{N}$.

# 3. CONFIDENTIALITY-PRESERVING RANGE QUERIES

In this section, we illustrate how to prevent the adversary from accessing data stored on $\mathcal{M}$ while ensuring efficient range-query processing and query-result authentication. The discussion on que ry-result completeness is deferred to § 4.

To ensure data confidentiality against $\mathcal{M}$, it is necessary to store encrypted data at $\mathcal{M}$ for which $\mathcal{M}$ has no decryption keys. For this purpose, node $S_i, \forall i \in [1, N]$, is preloaded with a distinct initial key $K_{i,0}$ uniquely shared with the network owner. At the end of epoch $t \geq 1$, $S_i$ generates an epoch key by $K_{i,t} = h(K_{i,t-1})$ and erases $K_{i,t-1}$ from its memory, where $h(\cdot)$ denotes a good hash function. Such epoch keys are used to realize forward-secure encryption of data produced in each epoch. For example, suppose that the adversary compromises $\mathcal{M}$ and $S_i$ during epoch $t + 1$. He will not be able to read the encrypted data $S_i$ submitted to $\mathcal{M}$ in the past $t$ epoches, as all the encryption (or decryption) keys $\{K_{i,x}\}_{x=1}^{t}$ no longer exist in $S_i$.

The most straightforward approach for data confidentiality is to let $S_i$ encrypt all the data generated during epoch $t$ as a whole before sending them to $\mathcal{M}$. Since $\mathcal{M}$ does not know $K_{i,t}$, it cannot read the data. Although providing strong confidentiality, this method fails to support efficient query processing. In particular, $\mathcal{M}$ need return to the network owner all the encrypted data it collected from $\{S_i\}_{i=1}^{N}$ in the specified epoch, as it cannot locate the encrypted data items exactly matching the query. The network owner can then derive all the epoch keys to decrypt the encrypted data and locate the data of interest if any. Since the network owner may only have interest in the data falling in narrow ranges, this method is obviously very inefficient especially when data retrieval is an expensive process, e.g., through an on-demand satellite link.

We adapt the technique in [10, 11] to strike a balance between data confidentiality and query efficiency. More specifically, the domain of attribute $A_j, \forall j \in [1, \mathbf{d}]$, is divided into $\omega_j \geq 1$ consecutive non-overlapping intervals under a public partitioning rule known to the master node and all the sensor nodes, sequentially numbered from 1 to $\omega_j$. A $\mathbf{d}$-dimensional bucket is defined by a tuple, $V = \langle v_1, v_2, \cdots, v_{\mathbf{d}} \rangle$ (called *bucket ID* hereafter), where $v_j \in [1, \omega_j], j \in [1, \mathbf{d}]$, is the interval index of $A_j$. Although it is possible that $\omega_i \neq \omega_j$ for $i \neq j$, we hereafter assume that $\omega_j = \omega, \forall j \in [1, \mathbf{d}]$, to simplify the presentation. When node $S_i$ produces some data falling into some bucket, we say that $S_i$ generated that bucket. We also denote by $Y_i$ the number of buckets $S_i$ generated during epoch $t$ and by $V_{i,j}, j \in [1, Y_i]$, the $j$th bucket ID.

At the end of epoch $t$, every node in cell $C$ encrypts all the data items falling into the same buckets as a whole and sends them with the corresponding bucket IDs to $\mathcal{M}$. Consider node $S_i$ as an example. Assume that $Y_i = 2$ and that $S_i$ has 3 and 2 data items in buckets $V_{i,1} = \langle 3, 5 \rangle$ and $V_{i,2} = \langle 6, 3 \rangle$, respectively. $S_i$ sends the following message to $\mathcal{M}$ at the end of epoch $t$:

$$S_i \rightarrow \mathcal{M} : i, t, \langle \langle 3, 5 \rangle, \{Data_1, Data_2, Data_3\}_{K_{i,t}} \rangle, \langle \langle 6, 3 \rangle, \{Data_4, Data_5\}_{K_{i,t}} \rangle,$$

where $\{\cdot\}_\star$ denotes an OCB-like authenticated encryption primitive [23] using the key on the subscript. For conciseness, let us denote all the data items in bucket $V_{i,j}$ by $D_{i,j}$. Then the above message can be represented as

$$S_i \rightarrow \mathcal{M} : i, t, \langle V_{i,1}, \{D_{i,1}\}_{K_{i,t}} \rangle, \langle V_{i,2}, \{D_{i,2}\}_{K_{i,t}} \rangle.$$

The query process is straightforward. The network owner first converts the desired data ranges (see Eq. (1)) into a set of bucket IDs, denoted by $\mathcal{Q}_t$, and then sends $\langle C, t, \mathcal{Q}_t \rangle$ to $\mathcal{M}$. Upon receipt of the query, $\mathcal{M}$ returns all the encrypted data buckets received during epoch $t$ whose IDs are within $\mathcal{Q}_t$ along with their corresponding sensor node IDs. The network owner can then derive all the corresponding epoch keys whereby to decrypt the received data buckets. Since the data ranges of interest may not exactly span consecutive full buckets, some buckets in the query reply may contain superfluous data items (false positives) the network owner does not want. One way to reduce such false positives and thus the communication overhead is to use finer bucketing, i.e., increasing $\omega$. This measure may, however, help $\mathcal{M}$ more accurately estimate the data distribution and thus jeopardize the data confidentiality to some extent. We refer to [11] for optimal bucketing strategies which can achieve a good balance between false positives and data confidentiality.

In addition to ensuring data confidentiality, the authenticated encryption primitive allows the network owner to detect forged or juggled data in the query result, as $\mathcal{M}$ does not know the correct epoch keys. Unfortunately, $\mathcal{M}$ may still omit data from some nodes which satisfy the query, leading to query-result incompleteness. This issue is tackled in the following section.

# 4. QUERY-RESULT COMPLETENESS VERIFICATION

In this section, we present a set of schemes for the network owner to verify the completeness of query results. Without loss of generality, we still consider cell $C$ with sensor nodes $\{S_i\}_{i=1}^{N}$ and a compromised (yet undetected) master node $\mathcal{M}$. For clarity only, we first temporarily ignore compromised sensor nodes and then discuss their impact in § 4.4.

To make subsequent theoretical analysis tractable, we make the following assumptions.

- There are totally $\mu$ (non-empty) data buckets generated in cell $C$ during each epoch, and each node $S_i$ on average produces $Y_i = \mu/N$ buckets.[1]
- A query about cell $C$ and epoch $t$ is represented by $\langle C, t, \mathcal{Q}_t \rangle$, where $\mathcal{Q}_t \subseteq \Omega$ denotes the set of queried bucket IDs and $\Omega = \{\langle v_1, v_2, \cdots, v_{\mathbf{d}} \rangle | v_i \in [1, \omega], i \in [1, \mathbf{d}]\}$. We further define $\gamma = |\mathcal{Q}_t|/\omega^{\mathbf{d}}$ as the ratio of queried bucket IDs among all the $\omega^{\mathbf{d}}$ ones.
- $\mathcal{M}$ omits each data bucket satisfying $\mathcal{Q}_t$ from the query response with equal *dropping probability* $\delta < 1$. Note that $\mathcal{M}$ may use various $\delta$s for different queries.

In what follows, we first present a verification scheme as a direct extension of the encoding technique [4] to multidimensional cases.

[1]We may ignore the rounding operation hereafter for simplifying the notation.

Then we introduce two novel probabilistic crosscheck schemes and further propose a random probing scheme to deal with compromised sensor nodes. The following two performance metrics will be used throughout.

- $P_{\text{det}}$**–detection probability**: the probability that $M$ is detected as having returned incomplete data.
- $\overline{\mathsf{T}}$**–communication cost**: the total communication energy consumption in bits resulting from completeness verification in cell $C$. Here we assume the same energy to transmit and receive each bit across each hop.

Note that we are only interested in minimizing the energy consumption of sensor nodes while ignoring the energy consumption for messages exchanged between master nodes and the network owner for two reasons. First, such message transmissions only involve master nodes which communicate via long-range, high-bandwidth radios. Second, master nodes are assumed to have plenty of energy (e.g., a solar panel and/or heavy-duty rechargeable batteries) than energy-constrained sensor nodes.

## 4.1 Completeness Verification based on Encoding Numbers

The basic idea of the encoding technique is to let each sensor node return some unforgeable proof for each empty bucket. Consider $S_i$ as an example. Let $\mathsf{V}_i = \{V_{i,j}\}_{j=1}^{Y_i} \subseteq \Omega$ denote the buckets $S_i$ generated in epoch $t$ and $D_{i,j}$ be the data items in bucket $V_{i,j}$. At the end of epoch $t$, $S_i$ generates a so-called *encoding number* for each empty bucket $V_{i,k} \in \Omega \setminus \mathsf{V}_i$ as

$$num(V_{i,k}, t) = h_{l_e}(i||t||V_{i,k}||K_{i,t}), \tag{2}$$

where $h_{l_e}(\cdot)$ denotes a good hash function of $l_e$ bits and $K_{i,t}$ is $S_i$'s epoch key. $num(V_{i,k}, t)$ is the proof that $S_i$ did not generate data in bucket $V_{i,k}$ during epoch $t$ and can be verified by the network owner who knows $K_{i,t}$. Finally, $S_i$ submits to $\mathcal{M}$ all the data buckets and encoding numbers as follows.

$$S_i \rightarrow \mathcal{M} : i, t, \{V_{i,j}, \{D_{i,j}\}_{K_{i,t}} | V_{i,j} \in \mathsf{V}_i\}, \\ \{V_{i,k}, num(V_{i,k}, t) | V_{i,k} \in \Omega \setminus \mathsf{V}_i\}.$$

Upon receiving the query $\langle C, t, \mathcal{Q}_t \rangle$, $\mathcal{M}$ should generate a hash of all the concatenated encoding numbers from all the $N$ nodes with corresponding empty bucket IDs in $\mathcal{Q}_t$.

$$\text{NUM}_{\mathcal{Q}_t} = h_{l_c}(\mathop{||}_{V_{i,k} \in \mathcal{Q}_t \cap \Omega \setminus \mathsf{V}_i, i \in [1,N]} num(V_{i,k}, t)),$$

where $h_{l_c}(\cdot)$ denotes a good hash function of $l_c$ bits. Then $\mathcal{M}$ returns all the data buckets satisfying $\mathcal{Q}_t$ along with $\text{NUM}_{\mathcal{Q}_t}$.

On receiving the response, the network owner can infer the empty buckets of each node in cell $C$. Since it knows all the epoch keys $\{K_{i,t}\}_{i=1}^{N}$, it can proceed to generate all the corresponding encoding numbers whereby to recompute $\text{NUM}_{\mathcal{Q}_t}$. If the result matches what it received, the network owner considers $\mathcal{M}$ legitimate and malicious otherwise.

### 4.1.1 Detection probability

Assume that $\mathcal{M}$ chooses to omit some qualified data buckets from the query result. To escape the detection by the network owner, it has to return a correct $\text{NUM}'_{\mathcal{Q}_t}$ corresponding to the incomplete data. If the length $l_c$ of $\text{NUM}'_{\mathcal{Q}_t}$ is sufficiently long, the probability $2^{-l_c}$ of directly guessing a $\text{NUM}'_{\mathcal{Q}_t}$ is negligible. Since $\mathcal{M}$ does not know the epoch keys of the corresponding sensor nodes which are necessary to compute the encoding numbers, the only option left for it is to guess the $l_e$-bit encoding numbers of the omitted data buckets.

Now we derive $P_{\text{det},E}$, the probability that $\mathcal{M}$ can be detected. Since each of the $\mu$ buckets is queried with probability $\gamma$ and omitted with probability $\delta$, the total number of omitted buckets is approximately $\mu\gamma\delta$. In addition, $\mathcal{M}$ can guess the correct encoding number for each omitted bucket with probability $2^{-l_e}$. The network owner cannot detect $\mathcal{M}$ if all the $\mu\gamma\delta$ buckets' encoding numbers are guessed correctly, which happens with probability $2^{-l_e\mu\gamma\delta}$. Thus we have

$$P_{\text{det},E} = 1 - 2^{-l_e\mu\gamma\delta}. \tag{3}$$

If $l_e\mu\gamma\delta$ is sufficiently large, then the detection probability $P_{\text{det},E}$ will be very high. This encoding technique can thus be viewed as a deterministic approach.

### 4.1.2 Communication cost

The communication cost $\overline{\mathsf{T}}$ of this scheme is incurred by transmitting the encoding numbers to $\mathcal{M}$. Since there are totally $\mu$ non-empty buckets and $N\omega^{\mathbf{d}} - \mu$ empty buckets, $N\omega^{\mathbf{d}} - \mu$ encoding numbers along with the corresponding bucket IDs need to be transmitted. Assuming that the average number of hops between each sensor node and $\mathcal{M}$ is $L_{avg}$, then $\overline{\mathsf{T}}$ is given by

$$\overline{\mathsf{T}}_E = (N\omega^{\mathbf{d}} - \mu)(l_e + \lceil \log \omega \rceil \mathbf{d})L_{avg}, \tag{4}$$

where $\lceil \log \omega \rceil \mathbf{d}$ is the length of a bucket ID in bits. As we can see, $\overline{\mathsf{T}}_E$ is acceptable when $\mu \simeq N\omega^{\mathbf{d}}$, i.e., there are almost no empty buckets. In practice, however, $\mu \ll N\omega^{\mathbf{d}}$ in event-driven sensor networks. This means that the communication cost of the encoding technique exponentially increases with $\mathbf{d}$ (the number of data attributes), i.e., $\overline{\mathsf{T}}_E = O(\omega^{\mathbf{d}}\mathbf{d})$. It is thus necessary to seek other techniques with better communication efficiency to cope with resource-constrained sensor nodes.

## 4.2 Probabilistic Spatial Crosscheck

The encoding technique enables deterministic detection of every query result's incompleteness with a communication cost $\overline{\mathsf{T}} = O(\omega^{\mathbf{d}}\mathbf{d})$. If the network owner can tolerate a small number of incomplete responses before detecting $\mathcal{M}$, it is feasible to design some probabilistic verification schemes with much less energy consumption.

The key idea of probabilistic spatial crosscheck is to embed some relationships among data generated by different sensor nodes. If $\mathcal{M}$ omits part of the data in the response, the network owner can decide with certain probability that the query result is incomplete by inspecting the relationships among other returned data. This technique thus forces $\mathcal{M}$ to either return all the data satisfying the query or risk being caught. To enable spatial crosscheck, we introduce a *gossip phase* at the end of each epoch $t$, in which sensor nodes exchange information about their sensed data before sending them to $\mathcal{M}$. Then each node that has data to submit appends some randomly-chosen received information to its own data buckets and then sends encrypted buckets to $\mathcal{M}$.

In particular, during the gossip phase of epoch $t$, each node with data for submission, say $S_i$, broadcasts a gossip message within cell $C$ which contains each of the generated bucket IDs $\{V_{i,j}\}_{j=1}^{Y_i}$ with equal probability $p_b$. Here we do not require the gossip messages, which only contain bucket indexes, to be encrypted, but we do assume a suitable broadcast authentication protocol like multi-level $\mu$TESLA [24] to ensure their authenticity.

At the end of the gossip phase, $S_i$ receives a gossip message from every other node that generated data in epoch $t$ and thus knows the IDs of all the non-empty buckets which include its own ones and

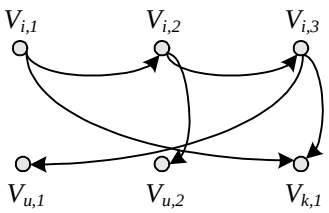



**Figure 2: A snapshot of the verification graph.**

are denoted by $\mathsf{V}$. Recall that $\{D_{i,j}\}_{i=1}^{Y_i}$ denote the data items in bucket $V_{i,j}$. Then $S_i$ appends each element in $\mathsf{V}$ along with the corresponding node ID to each element in $\{D_{i,j}\}_{i=1}^{Y_i}$ with equal probability $p_e$ with the exception that $V_{i,j}$ will not be appended to $D_{i,j}$. With this measure, it is possible that a bucket ID may be appended to multiple data blocks at other sensor nodes and also to other buckets generated by the same sensor node. Subsequently, $S_i$ sends encrypted data buckets to $\mathcal{M}$ as before.

For instance, suppose that $S_i$ generated data blocks $\{D_{i,j}\}_{j=1}^{3}$ in buckets $\{V_{i,j}\}_{j=1}^{3}$ (i.e., $Y_i = 3$), respectively, and received $V_{k,1}$ from node $S_k$ and $V_{u,1}$ and $V_{u,2}$ from node $S_u$. Also assume that $S_i$ have the following decisions.

- Append $\langle i, V_{i,2}\rangle$ and $\langle k, V_{k,1}\rangle$ to $D_{i,1}$;
- Append $\langle u, V_{u,2}\rangle$ and $\langle i, V_{i,3}\rangle$ to $D_{i,2}$;
- Append $\langle u, V_{u,1}\rangle$ and $\langle k, V_{k,1}\rangle$ to $D_{i,3}$.

Finally, $S_i$ sends the following message to $\mathcal{M}$.

$$\begin{aligned} S_i \rightarrow \mathcal{M} : i, t, &\langle V_{i,1}, \{D_{i,1}, \langle i, V_{i,2}\rangle, \langle k, V_{k,1}\rangle\}_{K_{i,t}}\rangle, \\ &\langle V_{i,2}, \{D_{i,2}, \langle u, V_{u,2}\rangle, \langle i, V_{i,3}\rangle\}_{K_{i,t}}\rangle, \\ &\langle V_{i,3}, \{D_{i,3}, \langle u, V_{u,1}\rangle, \langle k, V_{k,1}\rangle\}_{K_{i,t}}\rangle. \end{aligned}$$

Note that $\mathcal{M}$ cannot figure out the bucket IDs embedded in each bucket due to the encryption, that is, $\mathcal{M}$ does not know which buckets can crosscheck each other.

Upon receiving the query $\langle C, t, \mathcal{Q}_t\rangle$, $\mathcal{M}$ should return all the data buckets with IDs in $\mathcal{Q}_t$. The network owner then decrypts the data buckets in the response to get a set of embedded node/bucket ID pairs. If any such embedded bucket ID is in $\mathcal{Q}_t$ with the corresponding data bucket not being returned, the network owner decides that $\mathcal{M}$ omitted that data bucket and thus considers it malicious. For example, assume that $\mathcal{Q}_t$ contains $V_{i,1}$, $V_{i,2}$, and $V_{k,1}$ and that $\mathcal{M}$ did not return bucket $V_{k,1}$ of node $S_k$. If the network owner receives either of $\langle V_{i,1}, \{D_{i,1}, \langle i, V_{i,2}\rangle, \langle k, V_{k,1}\rangle\}_{K_{i,t}}\rangle$ and $\langle V_{i,3}, \{D_{i,3}, \langle u, V_{u,1}\rangle, \langle k, V_{k,1}, \rangle\}_{K_{i,t}}$, it can find that it should also have received bucket $V_{k,1}$ of node $S_k$. The network owner can then consider $\mathcal{M}$ malicious.

### 4.2.1 Detection probability

To analyze $P_{\det,S}$, we first define a *verification graph* which reflects the embedding relationships among all the $\mu$ data buckets generated in cell $C$ during epoch $t$.

*Definition 1. A* ***verification graph*** *is a directed graph $\mathcal{G} = (\mathsf{V}, \mathsf{E})$, where $\mathsf{V}$ is the set of vertexes with each being a non-empty bucket ID and $\mathsf{E}$ is the set of directed edges. An edge from vertex $V_{i_1,j_1}$ to vertex $V_{i_2,j_2}$ indicates that bucket $V_{i_1,j_1}$ contains $\langle i_2, V_{i_2,j_2}\rangle$ and thus can verify the existence of bucket $V_{i_2,j_2}$.*

A snapshot of $\mathcal{G}$ is given in Fig. 2, which corresponds to the example related to $S_i$ in § 4.2. For instance, the edge $\overrightarrow{V_{i,1}V_{k,1}}$ corresponds to the embedding relationship $\langle V_{i,1}, \{D_{i,1}, \langle i, V_{i,2}\rangle, \langle k, V_{k,1}\rangle\}_{K_{i,t}}\rangle$ whereby $V_{i,1}$ verifies the existence of $V_{k,1}$.

Since each of the $\mu$ data buckets satisfies $\mathcal{Q}_t$ with probability $\gamma$ and is omitted with probability $\delta$, the network owner receives on average $(1-\delta)\mu\gamma$ buckets. Let $\mathsf{V}_r$ and $\mathsf{V}_d = \mathsf{V} \setminus \mathsf{V}_r$ denote the set of vertexes (buckets) that the network owner receives and $\mathcal{M}$ omits, respectively. The network owner can detect the incompleteness of the query result as long as there is an edge $\overrightarrow{V_{i_1,j_1}V_{i_2,j_2}} \in \mathsf{E}$ such that $V_{i_1,j_1} \in \mathsf{V}_r$ and $V_{i_2,j_2} \in \mathsf{V}_d$. There are total $|\mathsf{V}_r||\mathsf{V}_d|$ possible edges between $\mathsf{V}_r$ and $\mathsf{V}_d$, where $|\mathsf{V}_r| = (1-\delta)\mu\gamma, |\mathsf{V}_d| = \delta\mu\gamma$. Referring to the aforementioned embedding process, we can see that an edge between any two vertexes in $\mathsf{V}$ exists with probability $p_b p_e$. The misbehavior of $\mathcal{M}$ cannot be detected if there is no edge from $\mathsf{V}_r$ to $\mathsf{V}_d$, which occurs with probability $(1-p_b p_e)^{|\mathsf{V}_r||\mathsf{V}_d|} = (1-p_b p_e)^{\mu^2\gamma^2\delta(1-\delta)}$. Therefore, we have

$$P_{\det,S} = 1 - (1-p_b p_e)^{\mu^2\gamma^2\delta(1-\delta)}. \tag{5}$$

### 4.2.2 Communication cost

Now we estimate the communication cost $\overline{\mathsf{T}}$, incurred by the gossip messages and the transmission of embedded node/bucket IDs to $\mathcal{M}$. To simplify the analysis, we assume the simplest broadcast technique with which each node receives and transmits a broadcast message once.

Recall that each bucket ID is of $\lceil \log \omega \rceil \mathbf{d}$ bits. Since each node on average produces $\mu/N$ buckets, $\mu p_b/N$ bucket IDs will be inserted into a gossip message. Assuming that each node ID is of $l_{id}$ bits, a gossip message is of $l_{id} + \mu p_b \lceil \log \omega \rceil \mathbf{d}/N$ bits. Since each of the $N$ gossip messages will be transmitted and received $N$ times, the associated communication cost is $N^2(l_{id} + \mu p_b \lceil \log \omega \rceil \mathbf{d}/N)$. According to the probabilistic gossiping and embedding process, each of the $\mu$ bucket IDs will have on average approximately $\mu p_b p_e$ copies at different nodes. Assuming that the average number of hops from a sensor node to $\mathcal{M}$ is $L_{avg}$, then the communication cost associated with transmitting the embedded node/bucket IDs to $\mathcal{M}$ is $\mu^2 p_b p_e L_{avg}(l_{id} + \lceil \log \omega \rceil \mathbf{d})$. Therefore, we have

$$\overline{\mathsf{T}}_S = N(N l_{id} + \mu p_b \lceil \log \omega \rceil \mathbf{d}) + \mu^2 p_b p_e L_{avg}(l_{id} + \lceil \log \omega \rceil \mathbf{d}). \tag{6}$$

Apparently, the communication cost of spatial crosscheck is of order $O(\mathbf{d})$, which is significantly better than that of the encoding technique, $O(\omega^{\mathbf{d}}\mathbf{d})$. This is achieved at a slight decrease in the detection probability, which will be shown in § 5.

## 4.3 Probabilistic Temporal Crosscheck

In this section, we propose a probabilistic temporal crosscheck technique as a complement to spatial crosscheck. The key idea is to embed some relationship between data buckets generated in consecutive epoches. If $\mathcal{M}$ did not correctly respond to query $\langle C, t, \mathcal{Q}_t\rangle$, but it answers query $\langle C, t, \mathcal{Q}_{t+1}\rangle$, it is possible that the network owner can catch $\mathcal{M}$ with very high probability. For the sake of clarity, we present the temporal crosscheck technique here and defer its integration with spatial crosscheck to § 4.5.

To enable temporal crosscheck, we request that node $S_i, \forall i \in [1, N]$, maintain a fixed-length buffer of $\mathcal{L}\lceil \log \omega \rceil \mathbf{d}$ bits, where $\mathcal{L} \leq \mu/N$. The buffer can thus hold $\mathcal{L}_i \leq \mathcal{L}$ bucket IDs, denoted by $\{B_{i,j}\}_{j=1}^{\mathcal{L}_i}$ and generated by $S_i$ in the last epoch. At the end of epoch $t$, $S_i$ appends with probability $p_t$ each element in $\{B_{i,j}\}_{j=1}^{\mathcal{L}_i}$ to each of the buckets $\{V_{i,j}\}_{j=1}^{Y_i}$ it generated in epoch $t$. Node $S_i$ also updates the buffer with $\mathcal{L}_i = \min\{\mathcal{L}, Y_i\}$ bucket IDs randomly chosen from $\{V_{i,j}\}_{j=1}^{Y_i}$.

As an example, assume that $\mathcal{L}_i = 3$ and that $S_i$ generated data blocks $\{D_{i,j}\}_{j=1}^{3}$ in buckets $\{V_{i,j}\}_{j=1}^{3}$ (i.e., $Y_i = 3$), respectively,

and makes the following decisions at the end of epoch $t$.

- Append $B_{i,1}$ and $B_{i,2}$ to $D_{i,1}$;
- Append $B_{i,3}$ to $D_{i,2}$;
- Append $B_{i,2}$ to $D_{i,3}$.

Finally, $S_i$ submits to $\mathcal{M}$ the following message.

$$S_i \rightarrow \mathcal{M} : i, t, \langle V_{i,1}, \{D_{i,1}, \langle B_{i,1}, B_{i,2}\rangle\}_{K_{1,t}}\rangle, \langle V_{i,2}, \{D_{i,2}, \langle B_{i,3}\rangle\}_{K_{1,t}}\rangle, \langle V_{i,3}, \{D_{i,3}, \langle B_{i,2}\rangle\}_{K_{1,t}}\rangle.$$

Note that $\mathcal{M}$ cannot figure out which data buckets contain any given $B_{i,j}$ due to the encryption.

Temporal crosscheck relies on queries for consecutive epoches. Consider two queries $\langle C, t, \mathcal{Q}_{t-1}\rangle$ and $\langle C, t, \mathcal{Q}_t\rangle$. The network owner can verify the completeness of the query result for $\mathcal{Q}_{t-1}$ based on the query result for $\mathcal{Q}_t$. If the decrypted result for $\mathcal{Q}_t$ contains any $B_{i,j}$ in $\mathcal{Q}_{t-1}$ for which the data bucket was not found in the query result for $\mathcal{Q}_{t-1}$, the network owner considers $\mathcal{M}$ malicious. For example, assume that $B_{i,1}$ was omitted in response to query $\mathcal{Q}_{t-1}$ and undetected, but $\langle V_{i,1}, \{D_{i,1}, \langle B_{i,1}, B_{i,2}\rangle\}_{K_{1,t}}\rangle$ is returned in response to query $\mathcal{Q}_t$. After doing the decryption, the network owner can find that $B_{i,1}$ satisfies $\mathcal{Q}_{t-1}$, but the corresponding data bucket was not received; $\mathcal{M}$ can thus be detected.

### 4.3.1 Detection probability

Recall that there are on average totally $\mu$ data buckets generated during epoch $t-1$. For simplicity, we assume that $Y_i = \mu/N \geq \mathcal{L}$ so that the buffer at each node is full of bucket IDs with each satisfying a query with probability $\gamma$. Since $\mathcal{M}$ omits each qualified bucket with probability $\delta$, there are totally $\mathcal{L}\gamma\delta$ bucket IDs in the buffer of each node, say $S_i$, whose data buckets are omitted in the response to $\mathcal{Q}_{t-1}$. If receiving a query $\langle C, t, \mathcal{Q}_t\rangle$, $\mathcal{M}$ will return $\mu\gamma(1-\delta)/N$ buckets of $S_i$ to the network owner. If none of them contains any omitted bucket ID in the buffer of epoch $t-1$, $\mathcal{M}$ cannot be detected as having omitted $S_i$'s bucket in the response to query $\mathcal{Q}_{t-1}$, which occurs with probability $(1-p_t)^{\mathcal{L}\mu\gamma^2\delta(1-\delta)/N}$. Since there are $N$ nodes, the network owner cannot detect the incompleteness of the result for query $\mathcal{Q}_{t-1}$ with probability $(1-p_t)^{\mathcal{L}\mu\gamma^2\delta(1-\delta)}$. Also note that $\mathcal{M}$ cannot be detected either if the network does not issue a query $\mathcal{Q}_t$, which is assumed to occur with probability $1-p_q$. It follows that

$$\begin{aligned} P_{\mathrm{det},T} &= 1-(1-p_q)-p_q(1-p_t)^{\mathcal{L}\mu\gamma^2\delta(1-\delta)} \\ &= p_q(1-(1-p_t)^{\mathcal{L}\mu\gamma^2\delta(1-\delta)}). \end{aligned} \quad (7)$$

### 4.3.2 Communication cost

The communication cost $\overline{\mathsf{T}}$ of temporal crosscheck is incurred by transmitting the embedded bucket IDs from the buffer at each of the $N$ nodes. Each node has on average $\mu/N$ buckets for submission to $\mathcal{M}$, into each of which $\mathcal{L}p_t$ bucket IDs from the buffer will be inserted. Assuming that the average number of hops from each node to $\mathcal{M}$ is $L_{avg}$, we thus have

$$\overline{\mathsf{T}}_T = \mathcal{L}\mu p_t \lceil \log \omega \rceil \mathbf{d} L_{avg}, \quad (8)$$

which is of order $O(\mathbf{d})$.

## 4.4 Impact of Compromised Sensor Nodes

We have temporarily ignore the impact of compromised sensor nodes to ease the presentation. In practice, however, the adversary may additionally compromise some sensor nodes among $\{S_i\}_{i=1}^N$ to help $\mathcal{M}$ escape detection. Compromised sensor nodes will fully collaborate with $\mathcal{M}$, e.g., by revealing their epoch keys. As mentioned in § 3, $\mathcal{M}$ cannot recover the past epoch keys of compromised sensor nodes, so it cannot read or fabricate the data generated in the past epoches. This means that our schemes can provide forward-secure data confidentiality and authentication. In this section, we discuss the impact of compromised sensor nodes on query-result completeness verification and present some defenses.

**Case 1: Return wrong data.**

Compromised sensor nodes may return wrong data to the network owner through $\mathcal{M}$. For example, they can submit arbitrary data to $\mathcal{M}$ which is both encrypted and authenticated. Consequently, the network owner cannot immediately identify that they were actually forged by compromised sensor nodes. It is out of the scope of this paper to differentiate such wrong data from true data.

**Case 2: Disobey the verification operations.**

Both the encoding technique and spatial/temporal crosscheck rely on sensor nodes to provide some verification information for query-result completeness verification. Compromised sensor nodes can cooperate with $\mathcal{M}$ to misbehave as follows.

- For the encoding technique, $\mathcal{M}$ can use the known epoch keys to derive all the encoding numbers for the compromised nodes and omit arbitrary qualifying buckets from them in the query result without being detected.
- For spatial crosscheck, compromised sensor nodes can neither broadcast their own bucket IDs nor insert any received bucket ID into their own data buckets during the gossip phase.
- For temporal crosscheck, compromised sensor nodes can simply choose not to embed any bucket ID in the buffer into their newly generated data buckets.

We note that the above misbehavior will not affect the operations of non-compromised sensor nodes; therefore, the detection performance of the proposed techniques will not be affected much as long as non-compromised sensor nodes are always the majority. In particular, assume that a fraction $p_c \ll 0.5$ of sensor nodes $\{S_i\}_{i=1}^N$ are compromised. It amounts to considering a cell with entirely $N' = (1-p_c)N$ non-compromised nodes which together produce on average $\mu' = \mu(1-p_c)$ buckets in each epoch. All the previous descriptions and analysis of the detection probability and the communication cost still hold after replacing $N$ and $\mu$ with $N'$ and $\mu'$, respectively.

**Case 3: Omit data only from non-compromised sensor nodes.**

$\mathcal{M}$ may also defeat spatial and temporal crosscheck by returning data only from compromised sensor nodes. Since the buckets of compromised sensor nodes do not include the bucket IDs of non-compromised ones, the network owner cannot detect that $\mathcal{M}$ omitted the data of non-compromised sensor nodes.

We further propose a ***random probing*** scheme as a defense, in which the network owner probes some random nodes of which no data were returned in the query result. To enable this, node $S_i, \forall i \in [1, N]$, maintains a buffer of $\tau\mathcal{L}_R\lceil\log\omega\rceil\mathbf{d}$ bits, where $\tau$ and $\mathcal{L}_R$ are two system parameters. Each buffer can hold up to $\tau\mathcal{L}_R$ bucket IDs. At the end of each epoch $t$, $S_i$ replaces $\min\{\mathcal{L}_R, Y_i\}$ oldest bucket IDs in the buffer with $\min\{\mathcal{L}_R, Y_i\}$ ones randomly chosen from $\{V_{i,j}\}_{j=1}^{Y_i}$ generated in epoch $t$.

As an example, consider a random probing for query $\mathcal{Q}_t$ that was issued in epoch $t'$. After receiving the result for $\mathcal{Q}_t$, the network owner identifies the set $\mathcal{S}_t$ of nodes whose data are included in the query result. Then the network owner randomly picks $\lambda' = \min\{\lambda, N - |\mathcal{S}_t|\}$ nodes from $\{S_i\}_{i=1}^N \setminus \mathcal{S}_t$ whose data are not included in the query result, where $\lambda$ is a system parameter. Let

$\Lambda$ denote the IDs of the nodes to be probed. The network owner sends $\langle C, t, \Lambda\rangle$ to $\mathcal{M}$ which in turn sends a probe message $\langle i, t\rangle$ to $S_i, \forall i \in \Lambda$. On receiving the probe, $S_i$ returns $\langle i, \{t, \mathcal{B}_{i,t}\}_{K_{i,t'}}\rangle$ via $\mathcal{M}$ to the network owner, where $\mathcal{B}_{i,t}$ denote all the bucket IDs of epoch $t$ stored in its buffer and $K_{i,t'}$ is the epoch key of the current epoch $t'$. After receiving all the $\lambda'$ responses, the network owner decrypts them using the corresponding epoch keys. If any received bucket ID is in $\mathcal{Q}_t$, the network owner knows that $\mathcal{M}$ omitted the corresponding data bucket and thus considers $\mathcal{M}$ malicious.

$\tau$ determines the maximum number of epoches that a bucket ID can stay in the buffer. Here, we have implicitly assumed that the network owner will not delay the queries for epoch $t$ for more than $\tau$ epoches (i.e., $t' - t \leq \tau$). This is believed to be a valid assumption because the longer the query delay, the less useful the data might be. If this assumption does not hold (i.e., $t' > t + \tau$), the random probing technique does not work because all the bucket IDs of epoch $t$ no longer exist. Therefore, there is a tradeoff between the effectiveness of random probing and the storage cost of sensor nodes. Below we analyze the detection probability and the communication cost of random probing.

### 4.4.1 Detection probability

For simplicity, we assume that $\lambda' = \lambda$ and that $Y_i = \mu/N \geq \mathcal{L}_R, \forall i \in [1, N]$. Then the network owner will receive on average $\mathcal{L}_R$ bucket IDs in each of the $\lambda$ probe responses. $\mathcal{M}$ cannot be detected if none of the $\lambda\mathcal{L}_R$ bucket IDs is in $\mathcal{Q}_t$, which occurs with probability $(1-\gamma)^{\lambda\mathcal{L}_R}$. Therefore, $\mathcal{M}$ can be detected with probability

$$P_{\text{det},R} = 1 - (1-\gamma)^{\lambda\mathcal{L}_R}. \tag{9}$$

### 4.4.2 Communication cost

The communication cost of random probing is incurred by transmitting probe messages and responses between sensor nodes and $\mathcal{M}$. Assume that a probe message only contains the probed's ID and that the average distance between sensor nodes and $\mathcal{M}$ is $L_{avg}$ hops. Also let $l_{ep}$ denote the length of epoch IDs. We then have

$$\begin{aligned}\overline{\mathsf{T}}_R &= \lambda(l_{id} + l_{ep} + l_{id} + l_{ep} + \mathcal{L}_R\lceil\log\omega\rceil\mathbf{d})L_{avg} \\ &= \lambda(2l_{id} + 2l_{ep} + \mathcal{L}_R\lceil\log\omega\rceil\mathbf{d})L_{avg},\end{aligned} \tag{10}$$

which is of order $O(\mathbf{d})$.

## 4.5 Hybrid Crosscheck

It is natural to build a hybrid scheme on spatial crosscheck, temporal crosscheck, and random probing. Given that the above three techniques fail with probabilities $\overline{P_{\text{det},S}} = (1-p_b p_e)^{\mu^2\gamma^2\delta(1-\delta)}$, $\overline{P_{\text{det},T}} = (1-p_q)+p_q(1-p_t)^{\mathcal{L}\mu\gamma^2\delta(1-\delta)}$, $\overline{P_{\text{det},R}} = (1-\gamma)^{\lambda\mathcal{L}_R}$, respectively, $\mathcal{M}$ can thus be detected by the hybrid scheme with probability

$$P_{\text{det},H} = 1 - \overline{P_{\text{det},S}} \cdot \overline{P_{\text{det},T}} \cdot \overline{P_{\text{det},R}}\ . \tag{11}$$

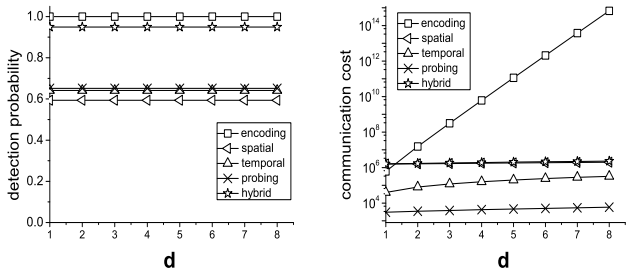
The communication cost of the hybrid scheme is simply given by $\overline{\mathsf{T}}_H = \overline{\mathsf{T}}_S + \overline{\mathsf{T}}_T + \overline{\mathsf{T}}_R$, where $\overline{\mathsf{T}}_S$, $\overline{\mathsf{T}}_T$ and $\overline{\mathsf{T}}_R$ are given in Eqs. (6), (8), and (10), respectively.

# 5. PERFORMANCE EVALUATION

In this section, we compare the detection probability $P_{\text{det}}$ and the communication cost $\overline{\mathsf{T}}$ of the proposed schemes using numerical results. We also did some simulations in which random network topologies were generated many times, and the simulation results are very close to the numerical results presented here and omitted due to space constraints. A high-level summary is given in Table 2.

**Table 1: Default Evaluation Parameters**

| Para. | Val. | Para. | Val. | Para. | Val. |
|---|---|---|---|---|---|
| $N$ | 400 | $\mathbf{d}$ | 2 | $\omega$ | 16 |
| $\mu$ | 1000 | $\gamma$ | 0.1 | $\delta$ | 0.1 |
| $p_b$ | 0.01 | $p_e$ | 0.1 | $p_q$ | 0.9 |
| $p_t$ | 0.5 | $p_c$ | 0 | $\mathcal{L}$ | 2 |
| $\mathcal{L}_R$ | 2 | $\lambda$ | 5 | $l_{id}$ | 10 bits |
| $l_e$ | 5 bits | $l_{ep}$ | 24 bits | $L_{avg}$ | 8 hops |




(a) detection probability $P_{\text{det}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 3: Impact of $\mathrm{d}$, the number of data dimensions.**

## 5.1 Numeric Results

We assume a cell consisting of 400 sensor nodes and a master node, and the average distance between a sensor node and the master node is 8 hops according to [25]. We also assume error-free and collision-free packet transmissions. Table 1 summarizes other default evaluation parameters unless specified otherwise.

### 5.1.1 Impact of $\mathbf{d}$

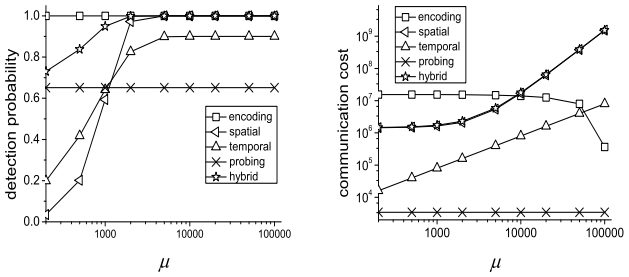
Fig. 3 shows the impact of $\mathbf{d}$, the number of dimensions or queriable attributes. We can see that the $P_{\text{det}}$s of the five schemes are all independent of $\mathbf{d}$. Under the default evaluation parameters, the encoding scheme has the highest detection probability $P_{\text{det},E} = 1$. Although the other schemes have smaller $P_{\text{det}}$s, they can still enable quick detection of $\mathcal{M}$. For example, since the detection probability of spatial crosscheck is $P_{\text{det},S} \approx 0.6$, the network owner can detect $\mathcal{M}$ after receiving $\lceil 1/P_{\text{det},S}\rceil = 2$ incomplete responses. The slight sacrifice in $P_{\text{det}}$ leads to significant savings in the communication cost $\overline{\mathsf{T}}$, which is shown in Fig. 3(b) in log 10 scale. Not surprisingly, the $\overline{\mathsf{T}}$ of the encoding scheme increases exponentially with $\mathbf{d}$, while the $\overline{\mathsf{T}}$ of each other scheme grows linearly with and is relatively insensitive to $\mathbf{d}$.

### 5.1.2 Impact of $\mu$

Fig. 4 shows the the impact of $\mu$, the total number of buckets generated in cell $C$ per epoch. Since the $P_{\text{det}}$ of the encoding scheme is still 1, we do not discuss it any further. The $P_{\text{det}}$s of spatial and temporal crosscheck dramatically increase as $\mu$ increases from 100 to 1000 and finally converge to $P_{\text{det},S} \approx 1$ and $P_{\text{det},T} \approx p_q = 0.9$, respectively. This observation is anticipated for two reasons. First, the larger $\mu$, the more buckets (i.e., $\mu\gamma\delta$) dropped by $\mathcal{M}$, and the more traces left for detection. Second, $P_{\text{det},T}$ is upper-bounded by $p_q$ (see Eq. (7)). Fig. 4(b) compares the $\overline{\mathsf{T}}$s of the five schemes. The encoding scheme's $\overline{\mathsf{T}}$ decreases with $\mu$ because the more data buckets generated, the fewer empty buckets and thus the fewer encoding numbers need be sent. In contrast, the $\overline{\mathsf{T}}$s of spatial crosscheck, temporal crosscheck, and the hybrid scheme all increase with $\mu$. This is not surprising because the more buckets generated, the more

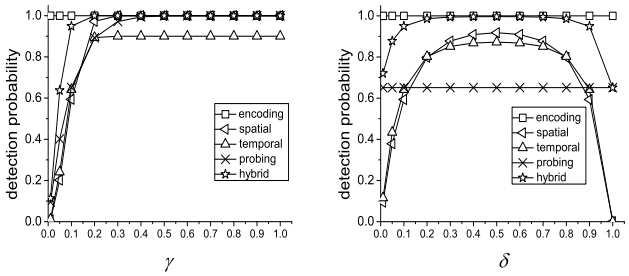
**Table 2: Comparison of Query-Result Completeness Verification Schemes**

| Scheme | Type | $P_{\text{det}}$ | $\overline{\mathsf{T}}$ | Limitation |
|---|---|---|---|---|
| **Encoding Number** | Deterministic | $\approx 1$ | $O(\omega^{\mathbf{d}}\mathbf{d})$ | High $\overline{\mathsf{T}}$ with large $\mathbf{d}$ |
| **Spatial Crosscheck** | Probabilistic | High when $\delta \approx 0.5$ | $O(\mathbf{d})$ | Ineffective when $\delta \to 1$ |
| **Temporal Crosscheck** | Probabilistic | High when $\delta \approx 0.5$ and $p_q \approx 1$ | $O(\mathbf{d})$ | Ineffective when $\delta \to 1$ or $p_q \to 0$ |
| **Random Probing** | Probabilistic | High but depends on query pattern | $O(\mathbf{d})$ | Affect by query delay |
| **Hybrid** | Probabilistic | Bounded by the best component | $O(\mathbf{d})$ | May fail only when all components fail |



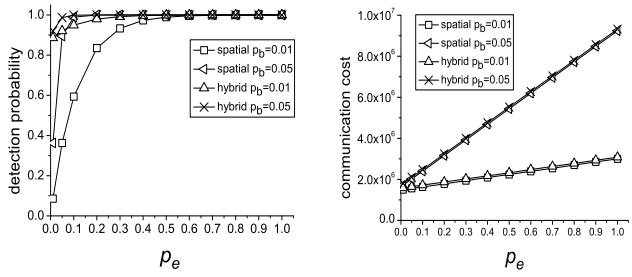

(a) detection probability $P_{\text{det}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 4: Impact of $\mu$, the total number of buckets per epoch.**



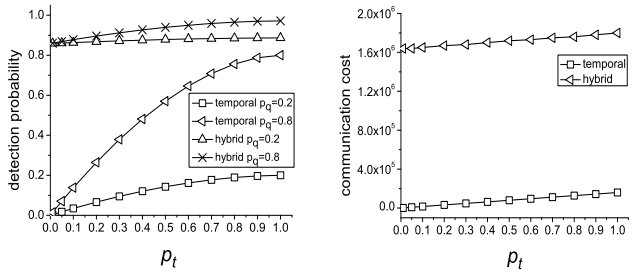

(a) detection probability $P_{\text{det}}$ (b) detection probability $P_{\text{det}}$

**Figure 5: Impact of $\gamma$ and $\delta$.**




(a) detection probability $P_{\text{det}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 6: Impact of $p_b$ and $p_e$.**




(a) detection probability $P_{\text{det}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 7: Impact of $p_q$ and $p_t$.**

relationships embedded among them. In practice, however, if we have some prior knowledge about $\mu$, we can easily tune the related parameters to reduce $\overline{\mathsf{T}}$, as Fig. 4(a) shows that it is unnecessary to have too many embedded spatial and/or temporal relationships. It is worth noticing that the random probing scheme has a much smaller $\overline{\mathsf{T}}$ with a moderate $P_{\text{dec}}$, which seems to outperform all the other four schemes. However, as we mentioned earlier, its effectiveness is also determined by the network owner's query pattern or the additional storage cost of sensor nodes. Therefore, we only consider it as a complementary scheme.

### *5.1.3 Impact of $\gamma$ and $\delta$*

Fig. 5(a) shows the impact of $\gamma$, the interest ratio or the probability of a bucket being queried. As we can see, the $P_{\text{det}}$s of the four probabilistic schemes increase with $\gamma$ because the larger $\gamma$, the more buckets received by the network owner, the easier to detect data incompleteness.

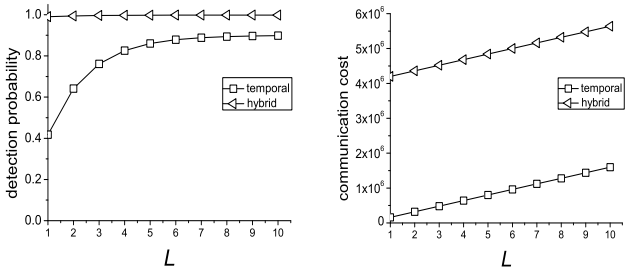
Fig. 5(b) shows the impact of $\delta$, the dropping probability. The $P_{\text{det}}$s of spatial and temporal crosscheck are both maximized at $\delta = 0.5$ and reach zero at $\delta = 1$ due to the term $\delta(1-\delta)$ in $P_{\text{det},S}$ and $P_{\text{det},T}$ (cf. Eq.(5) and Eq.(7)). This means that the spatial and temporal crosscheck schemes cannot detect $\mathcal{M}$ alone in all cases. The hybrid scheme overcomes this by integrating the random probing scheme whose $P_{\text{det}}$ is independent of $\delta$ because only the sensor nodes whose data do not appear in the response are probed. In addition, although the $P_{\text{det}}$ of the hybrid scheme is bounded by that of the random probing when $\delta$ approaches 1, the network owner can adjust $\lambda$ to achieve a high $P_{\text{det}}$.

### *5.1.4 Impact of $p_b$ and $p_e$*

Fig. 6 shows the impact of $p_b$ (the probability of a bucket ID being broadcasted) and $p_e$ (the probability of a received bucket ID being embedded) on the spatial and hybrid schemes. In general, the larger $p_b$ and $p_e$, the higher the $P_{\text{det}}$s, and the larger the $\overline{\mathsf{T}}$s. We can see in Fig. 6(a) that $p_b$ need not be too large; what really matters is $p_b p_e$ (cf. Eq. (5)). Fig. 6(b) shows that $p_b$ has larger influence on the $\overline{\mathsf{T}}$s of both schemes. Therefore, we can use smaller $p_b$ with larger $p_e$ to achieve a desirable $P_{\text{det}}$.

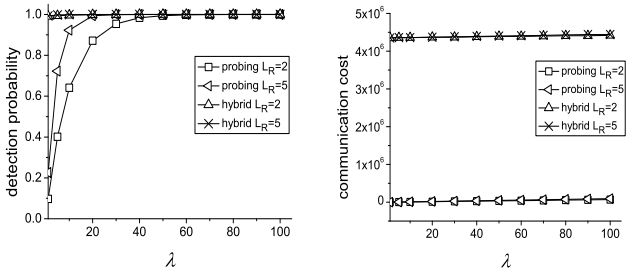
### *5.1.5 Impact of $p_q$ and $p_t$*

Fig. 7 shows the impact of $p_q$ (the probability that the network owner issues a query for a particular epoch) and $p_t$ (the probability that a bucket ID is stored in the buffer for temporal crosscheck) on temporal crosscheck and the hybrid scheme. In general, the larger $p_q$ and $p_t$, the higher $P_{\text{det}}$s, the larger $\overline{\mathsf{T}}$s. However, as we can see in Fig. 7(a), since the $P_{\text{det}}$ of temporal crosscheck is upper-



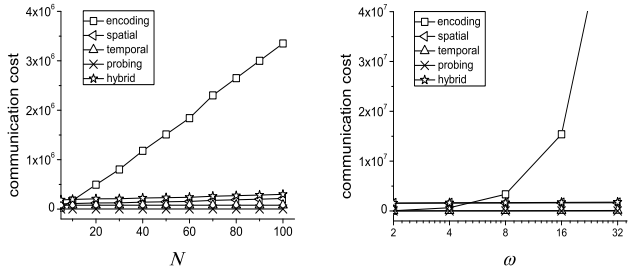

(a) detection probability $P_{\text{det}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 8: Impact of $\mathcal{L}$.**



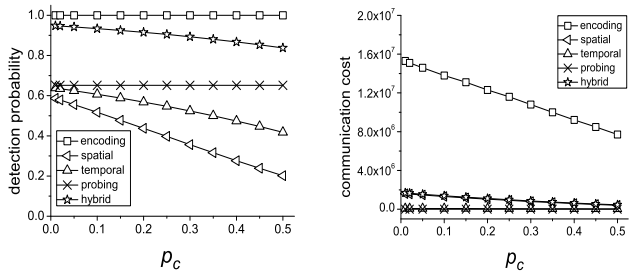

(a) detection probability $P_{\text{det}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 9: Impact of $\mathcal{L}_R$ and $\lambda$.**




(a) communication cost $\overline{\mathsf{T}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 10: Impact of $N$ and $\omega$.**




(a) detection probability $P_{\text{det}}$ (b) communication cost $\overline{\mathsf{T}}$

**Figure 11: Impact of $p_c$.**

bounded by $p_q$, the temporal scheme is less effective when $p_q$ is small. In contrast, $p_q$ and $p_t$ have less impact on the $P_{\text{det}}$ of the hybrid scheme whose detection probability is guaranteed by the other two components. In addition, there is a linear relationship between $p_t$ and the $\overline{\mathsf{T}}$ of temporal crosscheck, as shown in Fig. 7(b).

### 5.1.6 Impact of $\mathcal{L}$

Fig. 8 shows the impact of $\mathcal{L}$ (the buffer length) on the temporal crosscheck and hybrid schemes, where $\mu = 4000$ and $\gamma = 0.05$. As shown in Fig. 8(a), the larger $\mathcal{L}$, the higher the $P_{\text{det}}$s of both schemes, and vice versa. When $\mathcal{L} = 5$ or 6, the $P_{\text{det}}$ of the temporal scheme approaches its upper bound, $p_q$, as the term $(1-p_t)^{\mathcal{L}\mu\gamma^2\delta(1-\delta)}$ converges to 0 quickly with the increase of $\mathcal{L}$ (cf. Eq. (7)). This means that $\mathcal{L}$ need not be too large. In addition, the $\overline{\mathsf{T}}$ of the temporal scheme grows linearly with $\mathcal{L}$, as shown in Fig. 8(b).

### 5.1.7 Impact of $\mathcal{L}_R$ and $\lambda$

Fig. 9 shows the impact of $\mathcal{L}_R$ (the buffer length per epoch) and $\lambda$ (the number of probed nodes) on the random probing and hybrid schemes, where $\mu = 4000$ and $\gamma = 0.05$. As shown in Fig. 9(a), the larger $\mathcal{L}_R$ and $\lambda$, the higher $P_{\text{det}}$s for both schemes, and vice versa. When $\lambda$ is close to 30, the $P_{\text{det}}$s of both schemes approach 1. Therefore, $\mathcal{L}_R$ and $\lambda$ need not be very large, while $\tau$ need be large enough to tolerate possibly large delays between data generation and corresponding queries (§ 4.4). Fig. 9(b) shows that the $\overline{\mathsf{T}}$ of random probing is linear to $\lambda$, which is only a small portion of the $\overline{\mathsf{T}}$ of the hybrid scheme.

### 5.1.8 Impact of $N$ and $\omega$

Fig. 10(a) shows that the $\overline{\mathsf{T}}$ of the encoding technique grows linearly with $N \leq 2^{l_{id}}$ (the number of sensor nodes), while others' are almost unaffected. Fig. 10(b) shows the impact of $\omega$ (the number of buckets along one dimension), and we have the same observation as in Fig. 10(a).

### 5.1.9 Impact of $p_c$

Now we show the impact of $p_c$ (the probability that a sensor node is compromised). As shown in Fig. 11(a), the $P_{\text{det}}$s of both spatial and temporal crosscheck decrease with $p_c$, while the rest schemes remain almost unaffected. The reason is that the fewer buckets the network owner receives (§ 4.4), the more difficult to detect the master node's misbehavior. Fig. 11(b) shows the corresponding $\overline{\mathsf{T}}$s of all the schemes which all decrease with $p_c$. This is intuitive because we only care about the energy consumption of non-compromised sensor nodes and do not take compromised sensor nodes into account while computing $\overline{\mathsf{T}}$.

## 6. DISCUSSION OF FRAMING ATTACK

So far we have assumed that the adversary compromised the master node $\mathcal{M}$ and also some sensor nodes to aid $\mathcal{M}$. Our techniques allow the network owner to detect $\mathcal{M}$ with very high probability if it returned juggled and/or incomplete data. The adversary, however, may exploit our techniques to frame some legitimate master nodes. For example, some nodes in cell $C$ are compromised, but the master node $\mathcal{M}$ is non-compromised. The compromised sensor nodes can misbehave by submitting data to $\mathcal{M}$ which are encrypted and authenticated using incorrect epoch keys. They can also send to $\mathcal{M}$ completeness-verification information inconsistent with the submitted data. For example, for the encoding technique, compromised sensor nodes return incorrect encoding numbers to $\mathcal{M}$; for spatial and temporal crosscheck, they insert arbitrary bucket IDs into their data buckets for which the corresponding data buckets are actually not sent to $\mathcal{M}$; and for random probing, they can return fake bucket IDs in probe responses. Without precaution, such

misbehavior may mislead the network owner to falsely identify $\mathcal{M}$ as malicious. An effective defense against the framing attack is to let each sensor node and the master node $\mathcal{M}$ digitally sign each message transmitted and received. In case of dispute, the network owner can detect the compromised parties by analyzing the involved messages and signatures. Take random probing as an example. Assume that the compromised node $S_i$ returns bucket ID $V_{i,j} \in \mathcal{Q}_t$ in the probe response to the network owner. If $S_i$ can also present the signed proof from $\mathcal{M}$ that it has received the corresponding data bucket from $S_i$, the network owner considers $\mathcal{M}$ malicious; otherwise, $S_i$ is considered malicious. This solution involves public-key operations which, however, have been shown to be quite viable in sensor networks [26, 27]. Due to space constraints, we leave further investigation on the framing attack and the identification of compromised sensor nodes as future work.

## Acknowledgment

This work was supported in part by the US National Science Foundation under grant CNS-0716302. We would also like to thank anonymous reviewers for their constructive comments and helpful advice.

## 7. REFERENCES

[1] P. Desnoyers, D. Ganesan, and P. Shenoy, "TSAR: A two tier sensor storage architecture using interval skip graphs," in *ACM SenSys'05*, San Diego, California, USA, Nov. 2005, pp. 39–50.

[2] B. Sheng, Q. Li, and W. Mao, "Data storage placement in sensor networks," in *ACM MobiHoc'06*, Florence, Italy, May 2006, pp. 344–355.

[3] M. Shao, S. Zhu, W. Zhang, and G. Cao, "pDCS: Security and privacy support for data-centric sensor networks," in *IEEE INFOCOM'07*, Anchorage, Alaska, USA, May 2007, pp. 1298–1306.

[4] B. Sheng and Q. Li, "Verifiable privacy-preserving range query in sensor networks," in *IEEE INFOCOM'08*, Phoenix, AZ, Apr. 2008, pp. 46–50.

[5] 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 *ACM SenSys'06*, Boulder, Colorado, USA, Oct. 2006, pp. 153–166.

[6] X. Li, Y. J. Kim, R. Govindan, and W. Hong, "Multi-dimensional range queries in sensor networks," in *ACM SenSys'03*, Los Angeles, California, USA, Nov. 2003, pp. 63–75.

[7] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query processing system for sensor networks," *ACM Trans. Database Syst.*, vol. 30, no. 1, pp. 122–173, Mar. 2005.

[8] Y. Diao, D. Ganesan, G. Mathur, and P. J. Shenoy, "Rethinking data management for storage-centric sensor networks," in *CIDR'07*, Asilomar, CA, USA, Jan. 2007, pp. 22–31.

[9] J. Shi, R. Zhang, and Y. Zhang, "Secure range queries in tiered sensor networks," in *IEEE INFOCOM'09*, Rio de Janeiro, Brazil, Apr. 2009.

[10] H. Hacigümüş, B. Iyer, C. Li, and S. Mehrotra, "Executing SQL over encrypted data in the database-service-provider model," in *ACM SIGMOD'02*, Madison, Wisconsin, 6 2002, pp. 216–227.

[11] B. Hore, S. Mehrotra, and G. Tsudik, "A privacy-preserving index for range queries," in *VLDB'04*, Toronto, Canada, Aug. 2004, pp. 720–731.

[12] D. Liu, P. Ning, and W. Du, "Attack-resistant location estimation in sensor networks," in *IPSN'05*, Los Angeles, CA, Apr. 2005, pp. 99–106.

[13] Y. Zhang, W. Liu, Y. Fang, and D. Wu, "Secure localization and authentication in ultra-wideband sensor networks," *IEEE J. Select. Areas Commun., Special Issue on UWB Wireless Communications - Theory and Applications*, vol. 24, no. 4, pp. 829–835, Apr. 2006.

[14] W. Zhang, H. Song, S. Zhu, and G. Cao, "Least privilege and privilege deprivation: towards tolerating mobile sink compromises in wireless sensor networks," in *ACM MobiHoc'05*, Urbana-Champaign, IL, USA, May 2005, pp. 378–389.

[15] L. Eschenauer and V. Gligor, "A key-management scheme for distributed sensor networks," in *ACM CCS'02*, Washington, DC, Nov. 2002, pp. 41–47.

[16] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *IEEE S&P'03*, Oakland, CA, May 2003, pp. 197–213.

[17] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient security mechanisms for large-scale distributed sensor networks," in *ACM CCS'03*, Washington, DC, Oct. 2003, pp. 62–72.

[18] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *ACM CCS'03*, Washington, DC, Oct. 2003, pp. 52–61.

[19] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks," *IEEE J. Select. Areas Commun., Special Issue on Security in Wireless Ad Hoc Networks*, vol. 24, no. 2, pp. 247–260, Feb. 2006.

[20] L. Ma, X. Cheng, F. Liu, F. An, and M. Rivera, "iPAK: An in-situ pairwise key bootstrapping scheme for wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 8, pp. 1174–1184, Aug. 2007.

[21] R. Zhang, Y. Zhang, and K. Ren, "DP$^2$AC: Distributed privacy-preserving access control in sensor networks," in *IEEE INFOCOM'09*, Rio de Janeiro, Brazil, Apr. 2009.

[22] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and secure sensor data storage with dynamic integrity assurance," in *IEEE INFOCOM'09*, Rio de Janeiro, Brazil, Apr. 2009.

[23] P. Rogaway, M. Bellare, and J. Black, "OCB: A block-cipher mode of operation for efficient authenticated encryption," *ACM Trans. Inf. Syst. Secur.*, vol. 6, no. 3, pp. 365–403, Aug. 2003.

[24] D. Liu and P. Ning, "Multilevel $\mu$TESLA: Broadcast authentication for distributed sensor networks," *Trans. on Embedded Computing Sys.*, vol. 3, no. 4, pp. 800–836, 2004.

[25] L. E. Miller, "Distribution of link distances in a wireless network," *Journal of Research of the National Institute of Standards and Technology*, vol. 106, pp. 401–412, 2001.

[26] A. Liu and P. Ning, "TinyECC: A configurable library for elliptic curve cryptography in wireless sensor networks," in *IPSN'08*, St. Louis, MO, Apr. 2008, pp. 245–256.

[27] H. Wang, B. Sheng, C. C. Tan, and Q. Li, "Comparing symmetric-key and public-key based security schemes in sensor networks: A case study of user access control," in *ICDCS'08*, Beijing,China, Jun. 2008, pp. 11–18.