Computer Communications 108 (2017) 1–16

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

SER: Secure and efficient retrieval for anonymous range query in wireless sensor networks

Yao-Tung Tsou[a], Chun-Shien Lu[b,*], Sy-Yen Kuo[c]

[a] Department of Communications Engineering, Feng Chia University, 40724 No. 100, Wenhwa Rd., Seatwen, Taichung, Taiwan, ROC
[b] Institute of Information Science, Academia Sinica, 11529 No. 128 Academia Road, Sec. 2, Nankang, Taipei, Taiwan, ROC
[c] Department of Electrical Engineering, National Taiwan University, 10617 No. 1, Sec.4, Roosevelt Rd., Taipei, Taiwan, ROC.

ARTICLE INFO

Article history:
Received 27 April 2016
Revised 1 December 2016
Accepted 21 April 2017
Available online 3 May 2017

Keywords:
Anonymity
Privacy
Integrity
Range query
Wireless sensor network

ABSTRACT

A two-tiered wireless sensor network, where storage nodes take charge of storing sensitive data and processing queries with respect to the sensing nodes and querists, incurs a security breach. This is because the important role of storage nodes may be (1) instructed by attackers to return fake and/or incomplete data in response to querists' queries or (2) compromised by attackers to arbitrarily expose mass of valuable data or launch choice attacks to make authorized querists miss valuable data and disturb their decisions. Additionally, attackers may launch DoS attacks via wireless channels to storage nodes. To address the above issues, we propose a secure and efficient retrieval scheme for wireless sensor networks, namely SER, which mainly prevents attackers from gaining the valuable information stored on storage nodes, detects the compromised storage nodes when they misbehave, and efficiently verifies the querists' privileges without knowing their identities. In addition to privacy and system analyses, we demonstrate the feasibility and efficiency of SER through experiments conducted on TelosB prototype sensor platform equipped with IEEE 802.15.4 TI wireless transceiver, and conduct comparisons with state-of-the-art methods.

1. Introduction

Nowadays, large-scale wireless sensor networks (WSNs) are generally used in pragmatic applications for sensing and/or collecting information from monitored environment, whereby storage nodes take charge of aggregating and storing sensitive data, and processing queries with respect to the sensing nodes and querists. Among the large amount of data collected by sensing nodes, querists systematically derive the interested information from storage nodes to make decisions and take actions. For example, an army wants to collect information in a battlefield by launching a query to a storage node. Similarly, an oil company in order to locate the most viable oil reserve may want to launch a query to WSNs monitoring ocean atmosphere on the earth (GEOSS [1] and NOPP [2]).

Despite the advantage of scalable storage provided by current storage nodes, the stored data can be accidentally or maliciously disclosed to competitors or attackers by an untrusted storage node. Therefore, the reliance on untrusted storage nodes for data store and query processing raises severe security problems in hostile environments. In particular, storage nodes in military security or commercial service applications might be compromised by the attacker to maliciously destroy or expose valuable data. Also, such attacker can control a compromised storage node to return fake and/or incomplete data in response to querists' queries in order to disturb significant military or business decisions. These attacks are more subtle and harmful than passive eavesdropping, replay, and jamming attacks. To conquer the aforementioned attacks, a mechanism must not only protect the privacy of data stored on the storage nodes but also validate the integrity of query results. Apart from the issue of returning incomplete data in response to querists' queries, two potential attacks regarding user's data access privacy [3], *i.e.*, choice attacks[1] and Denial-of-Service (DoS) attacks, are also commonly encountered in the aforementioned scenarios. Storage nodes, when compromised, can facilitate choice attacks via unprotected identities to make the specific authorized querist miss valuable data and disturb his/her decisions. Moreover, storage nodes may endure DoS attacks such that they cannot execute querists' demands. In consequence of the above observations a se-

* Corresponding author.
E-mail address: lcs@iis.sinica.edu.tw (C.-S. Lu).

[1] Choice attack is where compromised nodes may not always launch their misbehavior. In other words, they only choose some specific querists to relay false and/or incomplete information.

http://dx.doi.org/10.1016/j.comcom.2017.04.007
0140-3664/

cure and efficient mechanism should not only ensure data privacy and integrity but also prevent storage nodes from suffering choice attacks and DoS attacks.

Intuitively, to ensure data privacy, sensing nodes can leverage a provably secure encryption mechanism to encrypt the data values before outsourcing them onto storage nodes. However, the first challenge is how a large volume of encrypted data stored on the storage nodes can be systematically and efficiently searched (or processed) for the queries (*e.g.*, range queries) launched by querists without decrypting the encrypted data values. The range query, by definition, is a typical sensor network operation that retrieves all data, where some values are between an upper and a lower boundaries. For data privacy, an encrypted data searchable scheme should be provided. Many studies [4–8] suggest that range query is a powerful type of query because collected/monitored data can be easily transformed into a data structure being applied for range query, *e.g.*, event/environment detection can be easily quantified as querying a range of the monitored variables associated with the event/environment. Furthermore, the encrypted data values stored on storage nodes may suffer from security breach. For example, the untrusted storage nodes can return the incomplete data to the querists or drive choice attacks to the specific querists. Thus, the second challenge is how to verify integrity of query result while untrusted storage nodes misbehave. The third challenge is how to prevent querists from suffering choice attacks while permitting authorized querists retrieving stored data from storage nodes. It should be noted that permitting querists accessing data from storage nodes would also raise security concern of DoS attacks. As a result, the fourth challenge is how to provide an anonymous access control which would resist DoS attacks.

In this paper, we focus on the above challenges and propose the corresponding solutions to address the aforementioned attacks. More precisely, we enable our method to resist eavesdropping, falsify, choice, and DoS attacks due to the following reasons. On one hand, due to inside attacks, the open nature of the sensor network makes the attacker passively eavesdrop even compromise the storage nodes to collect plenty of valuable data for mounting malicious behaviors to querists (such as omitting or tampering qualified data, inserting forged data, and driving choice attacks). On the other hand, due to outside attacks, the critical storage nodes may suffer unauthorized querists' attacks such as DoS attacks while providing capability of data access/query. Though inside and outside attacks are addressed separately in existing works [4,5,7–10], it is still insufficient and impractical in a two-tiered sensor network if they are not considered together. That is to say, only focusing on either secure range query or access control is vulnerable to compromised storage nodes. To seamlessly provide protections for sensitive data and querists' demands, we develop an efficient and secure retrieval scheme, called SER, with the capabilities of secure range query and anonymous access control.

Depending on the aforementioned applications of WSNs and challenges, there are four goals to achieve. First, from the aspect of data privacy, we should provide a searchable scheme to only permit storage nodes to process queries over the encrypted domain. Second, from the aspect of data integrity, storage nodes should submit verifiable information along with query results to querists since storage nodes may be malicious and misbehave. Third, from the aspect of access control, anonymous access control must be lightweight for preventing choice attacks from storage nodes and resisting DoS attack from querists at the same time. Last but not least, from the aspect of efficiency and effectiveness of utilization in WSN, a mechanism with less communication overhead and computation cost needs to be developed.

To the best of our knowledge, a secure and efficient retrieval scheme, featuring data privacy and integrity preservation (PIP) and anonymous access control (AAC) simultaneously, in the context of range query has not been developed.

1.1. Overview and contributions of this paper

SER efficiently provides not only provable encryption and integrity verifications for sensitive data privacy, but also secure and lightweight authentications for storage nodes and querists. More specifically, SER possesses an order encryption mechanism (OEM), which has the characteristics of the bucket scheme, to provide data privacy. The OEM technique allows a storage node to retrieve the desired data according to the stored sensitive data and queries over the encryption domain. In particular, the sophisticated OEM does not cause false positives (*i.e.*, does not generate useless data and send to querists) and disclose the information of bucket tags while the original design of bucket scheme does and is adopted in the renowned works [4,11,12]. To verify the integrity of query results, SER elaborates the data structure of Exclusive-OR Linked List (X2L), which simply processes the operation of Exclusive-OR (XOR), to construct neighborhood differences. For reducing communication cost, we extract the least significant bits (LSBs) of the neighborhood differences, which are operated with an innovative bit-map (BM) table, to verify the integrity of query results.

Also, SER develops a ring filter verification (RFV) strategy, which can achieve the goal of anonymous access control with concealing querists' identities and especially resisting DoS attacks. The RFV is a new concept of aiming to verify group privileges and resist DoS attacks simultaneously while the traditional ring verification or DoS attack resistance methods fail to achieve both and raise more overhead [9]. Through configuring the library of TinyECC and public keys of querists, we can make RFV be practical to a sensor network.

For clarifying the distinctions between some renowned methods and our proposed method, the comparisons, in terms of communication, computation, and storage overhead, are shown in Table 1. We can find that SER is more efficient than the prior works [4–6,10–13] while satisfying data privacy and integrity. In particular, our method, which can achieve both data privacy and integrity preservation (PIP) and anonymous access control (AAC), is more flexible and efficient in dealing with compromised storage nodes than *Priccess* [9]. In *Priccess*, it assumes that an attacker can launch both outside and inside attacks, implying that the attacker may eavesdrop the transmitted messages and compromise a number of sensor nodes subject to his/her choice (please refer to the Attacker Model of [9] for more details). Nevertheless, *Priccess* fails to deal with data privacy and access control that have been considered in SER.

The main contributions of this paper include:

- An original and provably secure OEM is proposed for systematically retrieving stored sensitive data over the encryption domain.
- An innovative BM table, operated with X2L, is proposed to guarantee the integrity of query results.
- A robust RFV scheme, which has capabilities to provide AAC for queries and resist DoS attacks for storage nodes, is developed.
- In comparison with the existing methods, our approach performs better in requiring less energy.

1.2. Organization of this paper

The rest of the paper is organized as follows. We give a comprehensive survey of related works in Section 2. Section 3 describes the system model and attacker model. Section 4 overviews Elliptic Curve Cryptography (ECC) and Constrained Function-based Authentication (CFA) that will be adopted in our method. We then describe the details of proposed method in Section 5. The formal data

Table 1
Comparisons of SER (our method) with state-of-the-art methods (N: the number of nodes; β: the number of sensitive data; γ: the number of consecutive non-overlapping regions; PIP: data privacy and integrity preservation; AAC: anonymous access control); EAV: resist eavesdropping attacks; FAL: resist falsify attacks; CHO: resist choice attacks; DoS: resist DoS attacks.

	Stored data security	Accurate query result	Detection probability	Communication overhead	Computation overhead	Storage overhead	PIP	AAC	EAV	FAL	CHO	DoS
Encoding approach (bucket scheme) [4]	Reasonable estimation	X	Always ≈ 1	$O(N)$	$O(\gamma)$	Exponential	V	X	V	V	X	X
Spatial crosscheck (bucket scheme) [11,12]	Reasonable estimation	X	Vary significantly	$O(N^2)$	$O(\gamma)$	Exponential	V	X	V	V	X	X
Temporal crosscheck (bucket scheme) [11,12]	Reasonable estimation	X	Vary significantly	$O(N^{1.5})$	$O(\gamma)$	Exponential	V	X	V	V	X	X
Hybrid crosscheck (bucket scheme) [11,12]	Reasonable estimation	X	Always ≈ 1	$O(N^2)$	$O(\gamma)$	Exponential	V	X	V	V	X	X
SafeQ [5]	Hard estimation	V	Always ≈ 1	$O(N)$	$O(\beta^2)$ (HMAC operations and/or Bloom filter generation)	Linear	V	X	V	V	X	X
SEF [13]	Hard estimation	V	Always ≈ 1	$O(N)$	$O(\beta \log \beta)$ (Hashing operations and AI tree generation)	Linear	V	X	V	V	X	X
QuerySec [6]	Hard estimation	V	Vary significantly	$O(N)$	$O(\beta)$	Linear	V	X	V	V	X	X
ESRQ [10]	Hard estimation	X	Always ≈ 1	$O(N)$	$O(\beta)$	Linear	V	X	V	V	X	X
SEMR [14]	Hard estimation	X	not provide	$O(N)$	$O(\beta)$	Linear	X	X	V	X	X	X
SecRQ [7]	Hard estimation	X	Always ≈ 1	$O(N)$	$O(\beta)$	Linear	V	X	V	V	X	X
CSRQ [8]	Hard estimation	X	Always ≈ 1	$O(N)$	$O(\beta)$	Linear	V	X	V	V	X	X
Priccess [9]	\	\	\	\	\	\	X	V	X	X	V	V
SER (our method)	Hard estimation	V	Always ≈ 1	$O(N)$	$O(\beta)$	Linear	V	V	V	V	V	V

privacy analysis is provided in Section 6, and the system analysis of other primitives is described in Section 7. Section 8 describes the performance evaluation, followed by the conclusions presented in Section 9.

2. Related work

In the aspect of PIP, researchers strive to prevent the attackers from obtaining information stored or processed on storage nodes. The most common solution is that each storage node is only allowed to process the sensitive data sensed by sensing nodes and the queries issued from querists over the encryption domain. Some drawbacks, however, still exist in these state-of-the-art methods. While data privacy and query result integrity of the range query in [4,11,12] can be preserved exquisitely, owing to the use of bucket partitioning proposed by [15], their query result completeness is achieved in different manners. Unfortunately, they share some common weaknesses: (1) The bucket scheme used in their approaches may cause false positives because of receiving useless data. To be more precise, querists would receive some useless data values. (2) If a storage node is compromised, it may leak some information to attackers, such as bucket tags. This would allow attackers to obtain a reasonable estimation on how much data falling into a range, the sensitive data of sensing nodes, and the queries of querists. The same drawback of information leakage is also existed in our previous work, SMQ [16]. Though a secure ordered bucketization scheme was proposed in [17] to address the leakage problem of bucket tags, the drawback of false positives still remains and leads to inevitable and additional communication cost.

In [5], SafeQ-Basic and its optimized version, SafeQ-Bloom, were proposed. Since SafeQ-Basic applies prefix membership verification, which needs to produce many hash values via MD5 or SHA-1, it would incur large communication, computation, and storage overhead. As for SafeQ-Bloom, it employs the Bloom filter to represent hash values in order to reduce communication and storage overhead; however, producing Bloom filter will further lead to rapidly increase in computation cost. The burst of computation cost makes SafeQ-Bloom not efficient and not suitable for WSNs. Moreover, SafeQ-Bloom will output inaccurate query results due to its use of Bloom filters [6].

To address the above problems, Jiajun *et al*[13]. have proposed a secure retrieval scheme, namely SEF. They employ the Order-Preserving Symmetric Encryption (OPSE) and an Authenticity & Integrity tree (AI tree) to preserve data privacy and integrity, respectively. However, OPSE does not preserve privacy due to the order-relation leakage and incurs extra computational overhead due to construction of AI tree.

As for recent works in [6,10,18], protocols are proposed that enable storage nodes to process queries correctly while prevent them from revealing both data and queries to sensors and sink, respectively. Nevertheless, these three methods only slightly improve over SafeQ but reserve some drawbacks the same as SafeQ in heavy communication cost and inevitable false positives.

Dong et al. [7,14] employ generalized inverse matrices and distance-based range query approach for the protection of data privacy in considering the collusion and probability attacks. Moreover, they propose a mutual verification scheme to verify the integrity of query results. However, false positive and false negative would be occurred in a range query due to the property of Bloom Filter while our method is a precise range query protocol. In particular, [14] would also suffer from falsify attacks.

CSRQ [8], in addition to support range query, employs an encoding mechanisms and encrypted constraint chain to preserve data privacy and query result integrity, respectively. However, CSRQ being similar to bucket scheme [4] would make a querist receive a number of useless data due to the number of factor in a constraint chain (τ).

More importantly, the aforementioned methods, in practice, may not work thanks to the storage nodes are vulnerable to choice attacks and DoS attacks. A potential solution to overcome choice attacks is that each querist hides his/her identity so that his/her identity cannot be associated with the data he/she requests, the times he/she accesses the sensitive data in the network, the nodes he/she queries, *etc*. For example, projects [1,2] that involve deploying large-scale WSNs to adaptively observe the ocean atmosphere on the earth may have sensitive data that would attract numerous querists from both public and private sectors, which range from individual querists to universities, government research centers, and businesses. A querist may wish to keep confidential whether he/she accessed the sensitive data, the data types he/she was interested in, or from which nodes he/she obtained the data, since the disclosure of such information may be used against his/her interest. Specifically, there are some research projects [9,19,20] that have been proposed to prevent attackers or unauthorized querists from accessing the sensitive data stored on storage nodes.

In [19], a model providing the access control for distributed querists under a realistic attacker model was proposed. Nevertheless, authors do not consider that, if the storage nodes are compromised, their system would greatly raise security threats in disclosing the stored sensitive data to attackers and violating the privacy of querists. Unlike the querist access control approach without guaranteeing the privacy of querists in [19], SER, [9] and [20] aim to provide blind query for guaranteeing the privacy of querists. In [9] and [20], blind signatures are exploited for hiding querists' identities. However, the network querist in [20] cannot sign a query because of its construction of signatures while the network querist in SER and [9] can do. Consequently, an attacker can easily intercept the tokens and impersonate authorized network querists to access data at storage nodes. Furthermore, blind signature mechanism applied in [20] is depended on RSA. Comparing to [9] and SER which choose ECC because ECC has a significant advantage over RSA due to its computational efficiency, small key size and compact signatures, [20] is more inefficient and fragile. Additionally, one of common weaknesses in [9,20] is that their methods are impractical to protect the sensor network without guaranteeing data privacy under outside and inside attacks. Another is that they are inefficient for an access control scheme when a dishonest querist mounts DoS attacks without exposing the identity. Specifically, the stored sensitive data might be leaked out and the authentication pattern might be vulnerable to DoS attacks. On the contrary, the above drawbacks are addressed in SER. As a consequence, SER is more flexible than [9] and [20].

3. System model and attacker model

In this section, the system model and attacker model will be formulated and described in detail.

3.1. System model

As shown in Fig. 1, our system consists of four kinds of roles: network owner, querist, storage node, and sensing node, where querist, storage node, and sensing node are set up in the user level, storage level, and sensing level, respectively. The network owner is able to deploy sensing nodes and storage nodes, and organize querists into groups. Note that the querists who want to access the network need to be authorized by the network owner first so the querists must register with the network owner before executing their privileges. To achieve access control, querists who have similar access privileges should be organized into the same group by the network owner as well. Meanwhile, the network owner

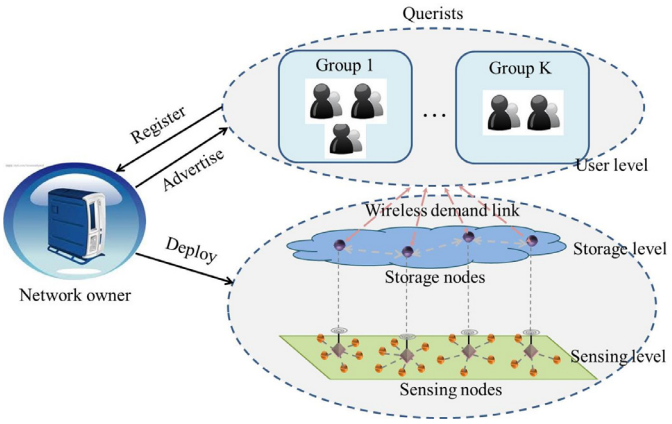


Fig. 1. Wireless sensor network with two-tiered architecture.

will advertise group identity and query privileges (types) for each group. Subsequently, the querists could send query commands to storage nodes according to the query types. In this paper, we consider range query command in the format $\{id, t, [e_a, e_b], \sigma\}$, where *id* denotes the designated sensing node's identity, t is the time slot the querist is interested in, $[e_a, e_b]$ represents the data value range of interest in the ciphertext form, and σ signifies the signature for blindly identifying the right of a querist.

On the other hand, the network is partitioned into physical cells by the network owner according to the geographic location, wherein each sensing node belongs to one cell, and each cell contains a storage node that is in charge of other sensing nodes in the cell. Depending on concrete applications, cells may overlap such that a sensing node in the overlapping region is affiliated with more than one storage node. In the user level, we follow the same assumption made in the prior works [4,5,9,11–13] that a querist is a powerful mobile device (*e.g.*, smart phone, laptop) equipped with much more storage capacity and power. In order to communicate with storage nodes, a querist is connected with a sensor (*e.g.*, TelosB). Then, the querist sends a query to storage nodes or gets the desired data from storage nodes via the sensor. The storage level can be viewed as a large-scale data center. Each storage node in the storage level collects the data sensed by the sensing nodes in the ground. Nevertheless, in order to achieve the goal of privacy preservation, storage nodes are only able to manipulate encrypted sensitive data. Moreover, sensing nodes are limited in storage, energy, and computation; they are responsible for sensing data, such as temperature, humidity, and the number of seismic events.

3.2. Attacker model

As sensors deployed in a hostile environment, attackers will try to overhear sensitive data from the network. The leakage of sensitive data breaches data privacy [21]. Moreover, attackers will incline to compromise storage nodes, which collect lots of sensitive data from the nearby sensing nodes. Doing so, the attackers can easily gain a large number of sensitive data or arbitrarily manipulate compromised storage nodes to return bogus and/or incomplete query results in response to queries from the querists. With assumption of compromising storage nodes, the range query format must be in ciphertext form to guarantee query privacy. Otherwise, (compromised) storage nodes can reasonably estimate what kind of data value they have or a querist is interested in according to query range. In addition, such storage nodes may, moreover, drive choice attacks to the specific querists. In contrast to the above threats, storage nodes also would suffer DoS attacks from the unauthorized querists. It is noted that choice attacks and DoS attacks are more general to the scenario of multiple querists considered in this paper.

Nevertheless, we do not consider the issue of compromised sensing nodes due to two reasons. First, without using any hardware protection methods, it is very difficult for network owner to prevent sensing nodes from being compromised. Second, data from a sensing node are a very small fraction of the entire data on a storage node.

4. Preliminaries

To make this paper self-contained, we briefly introduce ECC [22] and CFA [23], both of which will partially involve in the proposed RFV scheme.

4.1. Elliptic Curve Cryptography (ECC)

ECC [22] has been a popular security solution for wireless networks due to its small key size and low computational overhead. In particular, 160-bit ECC achieves the same security as 1024-bit RSA. An elliptic curve over a finite field $GF(q)$ (namely, a Galois Field of order q and q is a prime number) consists of a finite group of points (x_i, y_i), where $x_i, y_i \in Z$ satisfy:

$$y_i^2 = x_i^3 + ax_i^2 + b \ (mod\ q), \tag{1}$$

where a and $b \in GF(q)$.

Since the elliptic curve group operation has the property of closure, which means the addition of two points is still a point in the finite group over $GF(q)$, the rules for the addition over $GF(q)$ can be defined as follows. Given two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, the addition of these two points yields a point $R = (x_3, y_3)$, where $x_3, y_3 \in Z$ satisfy $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ and $L = (y_1 + y_2)/(x_1 + x_2)$ such that $x_3 = L^3 + L + x_1 + x_2 + a$ and $y_3 = L(x_1 + x_3) + x_3 + y_1$ [22]. Additionally, ECC relies on the difficulty of factoring or solving the elliptic curve discrete logarithm problem (ECDLP), that is, given P and Q in a finite group over $GF(q)$, it is difficult to compute a number k such that $Q = kP$.

Considering flexibility and efficiency of sensor networks, we will construct TinyECC [24] inside storage nodes and querists since TinyECC preserves the difficulty of factoring or solving ECDLP and has a configurable library so that it is suitable for resource-limited sensor platforms. With the configurable library of TinyECC, we also can turn optimizations of Public-key cryptography operations on or off based on the necessity during the authentication processes.

4.2. Constrained Function-based Authentication (CFA)

CFA is the first authentication scheme supporting en-route filtering with taking only a single packet of overhead. For incorporating CFA in our method, we construct the off-line step of CFA before sensor deployment and select a secret polynomial $f(x, y, z, w)$ from the constrained function set ξ, whose coefficients should be kept secret, thereby constituting the security basis of CFA. Please refer to [23] for the generation of the constrained function set ξ via the off-line procedure of CFA.

For simplicity, we assume that the degree of each variable in $f(x, y, z, w)$ is the same, although they can be distinct in our scheme. For each node u, the network planner constructs two polynomials, $f_{u,1}(y, z, w) = f(u, y, z, w)$ and $f_{u,2}(x, z, w) = f(x, u, z, w)$. Since directly storing these two polynomials enables the attacker to obtain the coefficients of $f(x, y, z, w)$ by capturing a few nodes, the authentication polynomial $auth_u(y, z, w)$ and verification polynomial $verf_u(x, z, w)$ should be constructed from the polynomials $f_{u,1}(y, z, w)$ and $f_{u,2}(x, z, w)$, respectively, by adding independent perturbation polynomials. Afterwards, the authentication and verification

polynomials are stored in node u. For source node u, the MAC attached to the message msg is calculated according to its own authentication polynomial. Let the verification number be the result calculated from the verification polynomial $verf_u(x, z, w)$ by applying the claimed source node ID, the shared pairwise key, and the hashed message into x, z, and w, respectively. The receiver considers the received message authentic and intact if and only if the verification difference, which is the difference between the MAC and its calculated verification number, is within a certain predetermined range.

5. Proposed SER scheme

In this section, we first introduce the techniques of OEM for preserving data (*i.e.*, sensitive data and queries) privacy, and RFV for AAC and DoS resistance. Then, we further describe the operations on sensing nodes, querists, and storage nodes.

5.1. Order Encryption Mechanism (OEM) for preserving data privacy

The concept of OEM is motivated from bucket scheme [4,11,12] which is an efficient way without resulting complex computation or additional communication overhead to allow data retrieval over encryption domain, comparing to sophisticated searchable encryption primitives [10,25,26]. Conversely, OEM does not inherit the shortcomings of inaccurate query results and bucket tags leakage of bucket scheme.

To ensure data privacy and enable storage nodes to retrieve inquired data over encryption domain, the design of encryption mechanism should be secure and reserve features that facilitate storage nodes to process encrypted data. As a consequence, we develop OEM that can be robust against attackers from chosen-plaintext attack (CPA) (proven in Section 6). The original design of OEM makes the characteristic of data ordering be known by sensing nodes and querists using consecutive non-overlapping regions. But, (compromised) storage nodes are not aware of the structure of OEM and therefore cannot know the information of order relations. In the following, we will illustrate the design of OEM in details.

OEM consists of two phases: order mapping and data encryption. In the order mapping phase, all data values (plaintexts) $\{d_i\}_{i=1}^{\beta}$ sensed by an sensing node with each of n bits are mapped to a region selected from an interval $[l_b, u_b]$. The length of the interval is chosen by the network owner to be proportional to the range of sensitive data. Then, the network owner partitions the interval for enquiry to γ ($\gamma > 0$) equivalent and consecutive non-overlapping regions with the order preserved. Each region r_ρ is associated with a sequence number ρ and a key k_ρ, $1 \le \rho \le \gamma$. In particular, the key k_ρ is one of parameters decided by the network owner in the initiation and is used to generate a keystream KS_ρ being a pseudorandom binary sequence. Here, the data value d_i mapped into a region r_ρ is denoted as d_i^ρ. Subsequently, each d_i^ρ ($1 \le i \le \beta$, $1 \le \rho \le \gamma$) will be encrypted using KS_ρ. It should be noted that k_ρ also works like an index so that a storage node can search encrypted data values in response to the queries of querists. More precisely, the indices are concealed from storage nodes in OEM so that storage nodes cannot conjecture what the stored sensitive data are by interpreting the indices. Oppositely, the bucket scheme used in [4,11,12] will expose the indices to storage nodes. We demonstrate in Fig. 2 the encryption phase consisting of a finite-state machine, to which we append an n-bit initialization vector (IV) to guarantee the semantic security.

In order to generate the KS_ρ with the size equal to the plaintext, an IV and k_ρ are used as the parameters of block cipher encryption. In particular, for the same plaintext that needs to produce different ciphertexts, a non-repeating counter (such as a non-repeating synchronized incremental counter [27]) is used as an IV,

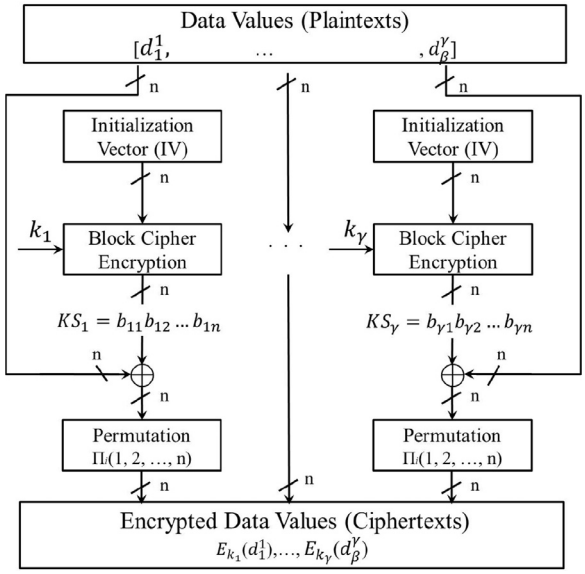


Fig. 2. Encryption of sensitive data.

which is involved as an input in the encryption operation. Thus, for the storage nodes, they store distinct encrypted sensitive data values in different time slots even though sensing nodes sense the identical data value. Therefore, the semantic security can be satisfied. For obtaining encrypted sensitive data values, each d_i^ρ ($1 \le i \le \beta$, $1 \le \rho \le \gamma$) is first XORed with a keystream KS_ρ. Subsequently, the resulting bits are permuted for improving security to obtain the encrypted data value $E_{k_\rho}(d_i^\rho)$ with inputting the key k_ρ as a parameter. The encryption processes are summarized in Algorithm 1 for clarity. Note that the same encryption mechanism

ALGORITHM 1: OEM process.

Input: the IV, $\{k_\rho\}_{\rho=1}^{\gamma}$, and sensitive data values $\{d_i\}_{i=1}^{\beta}$

Output: the encrypted data values $Enc_{k_\rho}(d_i^\rho)$ ($i \le \rho \le \gamma$, $1 \le i \le \beta$)

1. Map $\{d_i\}_{i=1}^{\beta}$ into γ equivalent and consecutive non-overlapping regions. For simplicity, the d_i mapped into the region r_ρ with the key k_ρ is termed as d_i^ρ.
2. Generate a set of keystream $\{KS_\rho\}_{\rho=1}^{\gamma}$ via the parameters IV and $\{k_\rho\}_{\rho=1}^{\gamma}$.
3. Process $d_i^\rho \oplus KS_\rho$ to hide the original number of '1's or '0's maximizes the entropy of the encrypted data item.
4. Permute the resulting bits of $d_i^\rho \oplus KS_\rho$ to improve security and generate the encrypted data values as the following:

$$Enc_{k_1}(d_1^1)|Enc_{k_1}(d_2^1)|\ldots|Enc_{k_\gamma}(d_\beta^\gamma), f \quad (2)$$

where "|" represents a concatenation operation.

is applied to encrypt queries so that storage nodes cannot learn what are the interests of a querist. For decryption, it works in the opposite way. Confirming the security strength of the encryption scheme, we will advance a proof in Section 6.

5.2. Ring Filter Verification (RFV) for AAC and DoS resistance

Because the original ring signature scheme [28] was not developed for anonymous access control in hostile environments, it cannot be directly applied to meet integrity protection of query demands, and availability of secure and efficient authentication for WSNs. Here, we design RFV to satisfy these requirements.

The intuition of elaborating RFV is to provide blind query authentication on storage nodes for all querists, and enable storage nodes to resist DoS attacks. Considering perfect forward secrecy and efficiency of authentication, we construct ring signature based on TinyECC, which is pragmatic and efficiently implemented in sensor networks [24], to hide querists' identities. Based on factoring [24], the construction of TinyECC is advantageous in terms of perfect forward secrecy, computational efficiency, small key size, and compact signature. On the contrary, traditional methods [29,30] face the challenges in inadequate security strength and high energy consumption. Unfortunately, a dishonest querist may also mount DoS attacks without exposing his/her identity due to the essence of ring signature. To remedy DoS attacks, we incorporate CFA with the resultant TinyECC-based ring signature as the latter has the benefit of low computational complexity. Our main goals are to achieve perfect forward secrecy and minimize the total power consumption of the nodes, and thereby save the precious sensor network energy. In other words, RFV can rapidly and efficiently verify querists' queries while preserving anonymity.

In order to explicitly construct TinyECC inside storage nodes and querists, we choose two field elements $a,\ b \in Z_q$ such that $E: y_i^2 = x_i^3 + ax_i + b \pmod q$ over Z_q and $4a^3 + 27b^2 \pmod q \neq 0$. Furthermore, setting $G(Z_q)$ to be the set of solutions $(x_i,\ y_i) \in Z_q \times Z_q$ to the congruence Eq. (1), $G(Z_q)$ is a finite group along with a point at infinity, denoted as O. Also, we choose a generator point $P' = (x_q, y_q)$ such that its order is a large prime number q over $G(Z_q)$, where $P' = O$. In this way, a subgroup G' of $G(Z_q)$ is constructed. Eventually, $H_1 : \{0,1\}^* \to Z_q^*$ and $H_2 : G' \times G' \to Z_q^*$ are chosen as cryptographic hash functions. It is worth noting that H_1 and H_2 will be public functions.

Every querist, say QT_j, initially creates its private key $x_j \in Z_q^*$ and public key $y_j = x_j \cdot P$, where P denotes a base point on the elliptic curve with order q. Then, QT_j publishes its public key. In particular, each storage node shall be pre-loaded with public keys prior to network deployment. The QT_j later could use authenticated broadcast, *e.g.*, μTESLA [31], to update storage nodes with a new public key when there is needed. By doing so, it is of efficiency in communication for advertising public keys. Aside from that, in order to conceal a querist's identity with lower computation overhead, the querist is able to randomly choose a portion of the querists' public keys and then create a TinyECC-based ring signature through his/her private key.

To resist DoS attacks, our efficient lightweight authentication scheme [23], CFA, is a good candidate. However, the primitive CFA supports data authentication without protecting data privacy and anonymity of queries for all querists at the same time. More precisely, query demands and querists' identities can be disclosed in the original CFA. So, we exploit CFA and incorporate it in the TinyECC-based ring signature, named RFV, to provide query demand privacy, enable querists to hide their identities, and enable storage nodes to resist DoS attacks.

In RFV, we suppose that there is a group of m querists, and their corresponding public keys form the set $\mathcal{Y} = \{y_1, \ldots, y_m\}$. To sign an encrypted query message $Q \in \{0,\ 1\}^*$ on behalf of the group, the legitimate signer (indexed by QT_u) employing the secret key x_u performs Algorithm 2 , while letting a querist QT_u send the encrypted query message Q to a storage node. QT_u performs the signature generation process as follows. First of all, the encrypted query message is signed by the querist via hash function H_1 and the base point P (Step 1). Then, the querist QT_u uses the generated group querists' public keys $y_1, \ldots, y_m$ and his/her private key $x_u \in Z_q^*$ to produce a signature σ for the said message Q (Steps $2-5$). At the same time, the querist QT_u would generate authentication number $MAC_u(v,\ Q)$ for the storage node v so that the storage node can verify the authenticity of encrypted query message

ALGORITHM 2: RFV signature procedure.

Input: encrypted query message Q
Output: signature σ

1 Sign encrypted query message (Q): $H_1(Q)P$.
2 Generate $\{a_i\}_{i=1}^m \in Z_q^*$ and compute $(x_i, y_i) = R_i = a_iP$ for each $i \in \{1, \ldots, m\}$, where $i \neq u$ and m represents group querists.
3 Randomly choose $v \in Z_q^*$, as well as compute R_u and $MAC_u(v, Q)$: $R_u = (x_u, y_u) = vP - \sum_{i=1, i\neq u}^m H_2(H_1(Q)P, R_i)y_i$,$MAC_u(v, Q) = auth_u(v, y_u, H_1(Q)P) + n_{u,s}$.
4 Compute $\alpha : \alpha = v + \sum_{i=1, i\neq u}^m a_i + x_u H_2(H_1(Q)P, R_u) \ (mod\ q)$.
5 Return $\sigma = \{R_1, \ldots, R_m, \mathcal{Y}, \alpha, MAC_u(v, Q)\}$.

Q (Step 3). Eventually, the querist QT_u sends the encrypted query message Q, combined with the signature σ, to the storage node v.

Upon the storage node receiving a query packet, including Q and σ, Algorithm 3 will be used to verify the rights of the

ALGORITHM 3: RFV verification procedure.

Input: encrypted query message Q and signature σ
Output: The result of verification

1 Compute $verf_v(u, y_v, H_1(Q)P)$.
2 Compute $VD_{v,u} = |verf_v(\cdot) - MAC_u(\cdot)|$.
3 Check whether $VD_{v,u}$ is within the range $\{0, \ldots, 2^{r-1} - 1\}$. If $VD_{v,u}$, in fact, is within the range $\{0, \ldots, 2^{r-1} - 1\}$, then goes to the next step; otherwise, the query will be rejected.
4 Compute $h_i : h_i = H_2(H_1(Q)P, R_i)$, $1 \leq i \leq m$.
5 Check the equation: $\alpha P = \sum_{i=1}^m (R_i + h_i y_i)$.

querist. The Steps $1-3$ in Algorithm 3 are used to verify the integrity of received message Q, which is considered authentic and intact if and only if the verification difference $VD_{v,\ u}$ is within a certain predetermined range $\{0, \ldots, 2^{r-1} - 1\}$. The Steps 4 and 5 are applied to authenticate the querist in accordance with received Q and σ, which is considered permissible if and only if $\sum_{i=1}^m (R_i + h_i y_i) = R_u + h_u y_u + \sum_{i=1, i\neq u}^m (R_i + h_i y_i) = vP + h_u y_u + \sum_{i=1, i\neq u}^m R_i = \alpha P$, where α is included in signature σ sent by the querist. Note that the notation "+" denotes addition of points over $G(Z_q)$.

5.3. Operations on sensing nodes, querists, and storage nodes

Based on the above setting, OEM and RFV are constructed and ready for use on sensing nodes, querists, and storage nodes. We describe the behaviors of sensing nodes, querists, and storage nodes in detail as follows. Without loss of generality, we focus on one cluster consisting of N sensing nodes denoted as $\{SN_i\}_{i=1}^N$, and a storage node. The OEM function is set on querists and $\{SN_i\}_{i=1}^N$ at system initialization. Also, the RFV function is set on a storage node and querists.

Sensing Node Behavior. To prevent a storage node from obtaining the sensitive data in plaintext, the sensing node applies OEM to encrypt sensitive data for confidentiality. To enable OEM, we map each n bits data value d_i, $1 \leq i \leq \beta$, sensed by a sensing node to a respective region. Each region will be assigned a key k_ρ, $1 \leq \rho \leq \gamma$, and k_ρ then is used to generate the corresponding n-bit keystream KS_ρ to encrypt the sensitive data mapped into that region. For example, if the interval [1,10] is partitioned into 5 equivalent and consecutive non-overlapping regions, then the key set $\{k_\rho\}_{\rho=1}^5$ generated by the key generator consists of 5 keys, k_1, k_2, k_3, k_4, and k_5. Considering the sensitive data set

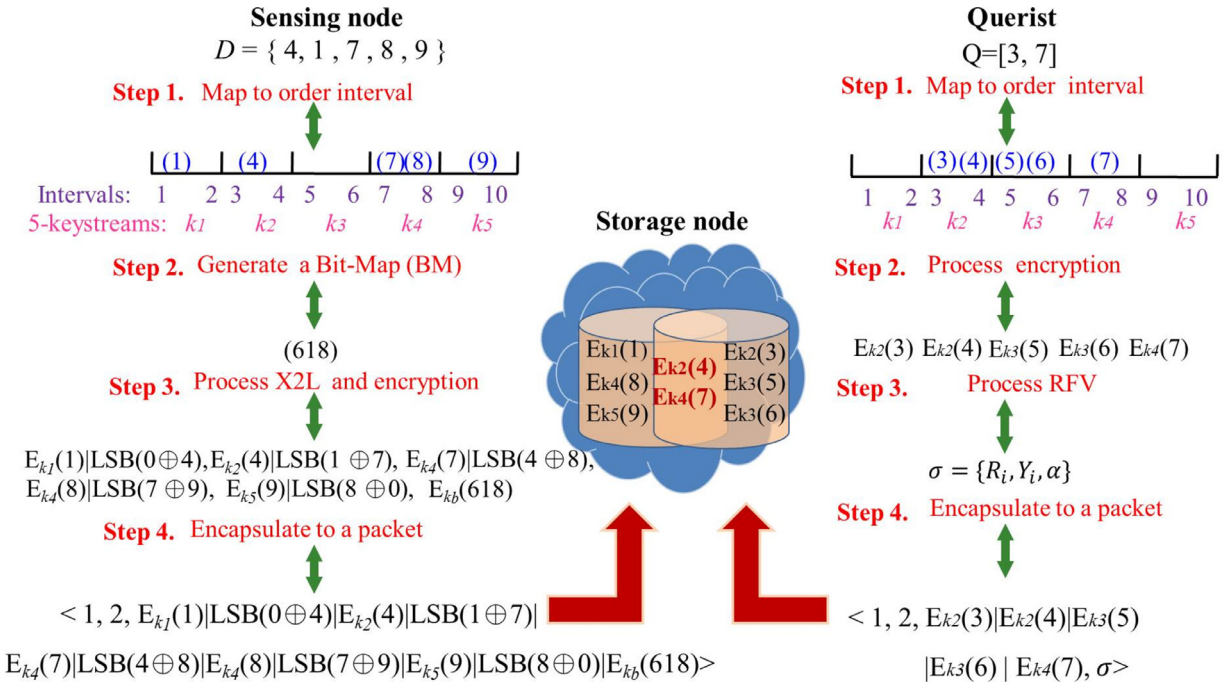


Fig. 3. Flowchart of data process.

$\{d_i\}_{i=1}^{5} = \{4, 1, 7, 8, 9\}$ collected by an SN_1 with $id = 1$ in time slot $t = 2$, we illustrate the behavior of the sensing node on the left-hand side of Fig. 3. The sensing node directly maps the sensitive data set $\{d_i\}_{i=1}^{5}$ to the ordered intervals (Step 1 on the left-hand side of Fig. 3) and encrypts them using XOR, namely randomized encoding in this paper, indicated in Fig. 2 (Step 3 on the left-hand side of Fig. 3).

As for the integrity of query results, we apply a data structure X2L to preserve and verify the integrity of a query result (Step 3 on the left-hand side of Fig. 3). The X2L is a data structure taking advantage of the bitwise XOR (denoted as $\oplus$ hereafter) operation to decrease storage requirements. More specifically, it compresses two data values into one by operating $d_{i-1} \oplus d_{i+1}$ as the neighborhood difference for the data value d_i. Because of its computational efficiency, we exploit X2L to preserve and verify the integrity of query result by concatenating neighborhood difference with each sensitive value. Given the data items $\{(1|d_0 \oplus 4), (4|1 \oplus 7), (7|4 \oplus 8), (8|7 \oplus 9), (9|8 \oplus d_{\beta+1})\}$ for the sensitive data set $\{d_i\}_{i=1}^{5} = \{4, 1, 7, 8, 9\}$ from a sensing node and the query range [3,7] for a question from a querist, where d_0 and $d_{\beta+1}$ are boundary values, as an example. We can recover the outliers of query range, 1 and 8, by calculating $1 \oplus 7 \oplus 7$ and $4 \oplus 4 \oplus 8$, respectively, using the query results $(4|1 \oplus 7)$ and $(7|4 \oplus 8)$. If the recovered outliers actually fall within the query range, the query results are considered as incompleteness. It is worth noting that X2L exploited in our method works like a chain, binding each data value together. When one of data values is lost or destroyed, the chain will be incomplete. For reducing the communication overhead, we only transmit the least significant bit of $d_{i-1} \oplus d_{i+1}$ (say, $LSB(d_{i-1} \oplus d_{i+1})$ or L_i hereafter) in the air. It is worth mentioning that each *LSB* needs to be operated with BM table to enable completeness verification, so the behavior of only maliciously modifying *LSB* values will be detected via completeness verification. In addition, in order to prevent a storage node from responding with an empty or false query result when actually there are sensitive data values that correspond to the query of querist, the sensing node constructs a BM table for querists to further check the incompleteness of a query result (Step 2 on the left-hand side of Fig. 3).

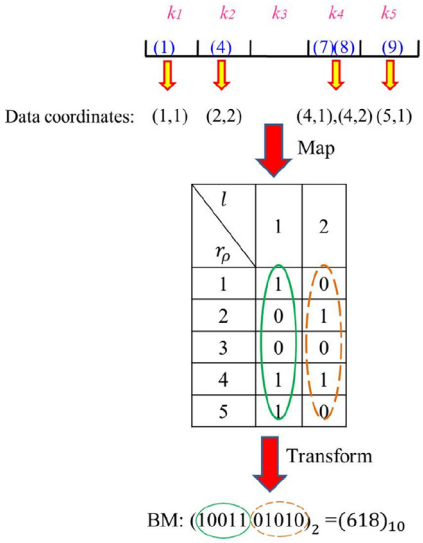


Fig. 4. Construction of a BM value.

Actually, BM is a table, which indexes the locations of sensitive data values fallen in the intervals, as illustrated in the Step 1 of Fig. 3. When the sensitive data are located within the intervals, the sensing node labels each data item as the form of (r_ρ, l), which represents that a data value falls into the lth location of the region r_ρ. As the example shown in Fig. 4, data coordinate (4,1) represents that a data value falls into the 1st location of the region $r_\rho = 4$ (indexed by k_4). Thus, the corresponding location $(r_\rho, l) = (4, 1)$ in the BM table will be marked by 1. Following the above rule, the five data values in Fig. 4 can be recorded in the BM table, and the relative binary string $(1001101010)_2$ will be transferred to a decimal number $(618)_{10}$, which is defined to be a BM or a BM value here. According to positions of data values recorded in the BM table and *LSB* values, querists can systematically verify the completeness of query results. The operations of completeness verifi-

cation using BM table will be described later in **Querist Behavior**. It should be noted that the BM represents the locations of actual data values; thus, querists cannot list all the actual values from the BM. They still need to query the storage nodes for responding the actual data values. Furthermore, BM is only outsourced in the ciphertext form to a storage node, so an attacker or the (compromised) storage node is not able to figure out encrypted BM without an encryption key. If the sensing values are located in the range of $[1, \Im]$, the size of BM is $\Im$ bits.

Ultimately, the SN_1 encapsulates its $id = 1$, time slot $t = 2$, encrypted data values, and an encrypted BM, denoted as $E_{k_b}(BM)$, into a packet expressed as:

$$\{1, 2, E_{k_1}(1)|LSB(0 \oplus 4)|E_{k_2}(4)|LSB(1 \oplus 7)|E_{k_4}(7)|LSB(4 \oplus 8) \\ |E_{k_4}(8)|LSB(7 \oplus 9)|E_{k_5}(9)|LSB(8 \oplus 0)|E_{k_b}(618)\},$$

and sends the packet to its closest storage node (Step 4 on the left-hand side of Fig. 3). Note that, for the encrypted data values in a packet (Eq. (3)), it is composed of β ciphertexts via concatenation, each of which has n bits. More specifically, the plaintext mentioned here is, in fact, composed of a sensitive plaintext and its corresponding least significant bit of neighborhood difference.

Querist Behavior. If a querist wants to retrieve some available sensitive data values from a storage node, he/she needs to apply OEM for generating the encrypted query message (Q) (Steps 1 and 2 on the right-hand side of Fig. 3). Then, the encrypted query message is signed by the querist via RFV (Algorithm 2) for hiding his/her identity and generating authentication number. After executing RFV, the obtained signature σ will be encapsulated with other query information (*i.e.*, sensing node's *id*, time slot, Q) to form the query packet $\{id, t, Q = [e_a, e_b], \sigma\}$. Subsequently, the query packet will be sent by the querist to a storage node.

Once receiving the response from the storage node, the querist will obtain a query result (QR) and a BM value by decomposing the ciphertext every $n+1$ bits in the packet via Eq. 3. At the same time, a BM table shall be reconstructed by the BM value. More specifically, converting the BM value to a binary string, we will store it in the BM table. For simplicity, we let the region size used in order mapping of OEM be τ and let the set of actual query results be $\{E_{k_{\lceil \frac{d_\ell}{\tau} \rceil}}(d_\ell^{\lceil \frac{d_\ell}{\tau} \rceil})|L_\ell,\ E_{k_{\lceil \frac{d_{\ell+1}}{\tau} \rceil}}(d_{\ell+1}^{\lceil \frac{d_{\ell+1}}{\tau} \rceil})|L_{\ell+1},\ \dots,\ E_{k_{\lceil \frac{d_\hbar}{\tau} \rceil}}(d_\hbar^{\lceil \frac{d_\hbar}{\tau} \rceil})|L_\hbar\}$, where $d_\ell^{\lceil \frac{d_\ell}{\tau} \rceil} < d_{\ell+1}^{\lceil \frac{d_{\ell+1}}{\tau} \rceil} < \cdots < d_\hbar^{\lceil \frac{d_\hbar}{\tau} \rceil}$. It should be noted that each encrypted item $(E_{k_{\lceil \frac{d_j}{\tau} \rceil}}(d_j^{\lceil \frac{d_j}{\tau} \rceil})|L_j)$, $\ell \le j \le \hbar$, can be decrypted to attain $(d_j^{\lceil \frac{d_j}{\tau} \rceil}|L_j)$ using a key $k_{\lceil \frac{d_j}{\tau} \rceil}$. Subsequently, there are four cases that the querist has to consider for integrity detection.

- If there exists $\ell < j < \hbar$ but $(d_{j-1}^{\lceil \frac{d_{j-1}}{\tau} \rceil}|L_{j-1}) \notin QR$, then the querist can know the missing $d_{j-1}^{\lceil \frac{d_{j-1}}{\tau} \rceil}$ by finding an available value $d_k^{\lceil \frac{d_k}{\tau} \rceil}(k \neq j)$, whose location in the BM table is less than or equal to $\lceil \frac{d_j}{\tau} \rceil$, then $d_k^{\lceil \frac{d_k}{\tau} \rceil} = d_{j-1}^{\lceil \frac{d_{j-1}}{\tau} \rceil}$ *s.t.* $d_k^{\lceil \frac{d_k}{\tau} \rceil}(k \neq j)$ satisfies $LSB(d_k^{\lceil \frac{d_k}{\tau} \rceil} \oplus d_{j+1}^{\lceil \frac{d_{j+1}}{\tau} \rceil}) = L_j$.
- If $(d_\ell^{\lceil \frac{d_\ell}{\tau} \rceil}|L_\ell) \notin QR$, then the querist can detect the missing $d_\ell^{\lceil \frac{d_\ell}{\tau} \rceil}$ by finding an available value $d_k^{\lceil \frac{d_k}{\tau} \rceil}(k \neq \ell + 1)$, whose location in the BM table is less than or equal to $\lceil \frac{d_{\ell+1}}{\tau} \rceil$, then $d_k^{\lceil \frac{d_k}{\tau} \rceil} = d_\ell^{\lceil \frac{d_\ell}{\tau} \rceil}$ *s.t.* $d_k^{\lceil \frac{d_k}{\tau} \rceil}(k \neq \ell + 1)$ satisfies $LSB(d_k^{\lceil \frac{d_k}{\tau} \rceil} \oplus d_{\ell+2}^{\lceil \frac{d_{\ell+2}}{\tau} \rceil}) = L_{\ell+1}$.
- If $(d_\hbar^{\lceil \frac{d_\hbar}{\tau} \rceil}|L_\hbar) \notin QR$, then the querist can detect the missing $d_\hbar^{\lceil \frac{d_\hbar}{\tau} \rceil}$ by finding an available value $d_k^{\lceil \frac{d_k}{\tau} \rceil}(k \neq \hbar - 1)$, whose location in the BM table is large than or equal to $\lceil \frac{d_{\hbar-1}}{\tau} \rceil$, then $d_k^{\lceil \frac{d_k}{\tau} \rceil} = d_\hbar^{\lceil \frac{d_\hbar}{\tau} \rceil}$ *s.t.* $d_k^{\lceil \frac{d_k}{\tau} \rceil}(k \neq \hbar - 1)$ satisfies $LSB(d_k^{\lceil \frac{d_k}{\tau} \rceil} \oplus d_{\hbar-2}^{\lceil \frac{d_{\hbar-2}}{\tau} \rceil}) = L_{\hbar-1}$.
- If QR only contains a BM value, then the querist only needs to recover the BM table and observe the data locations to further verify whether data values corresponding to the query of the querist exist or not.

Storage Node Behavior. A storage node's behavior contains three processes: preserving encrypted sensitive data, verifying rights, and retrieving encrypted sensitive data. They are, respectively, described as follows. We first describe how to preserve encrypted data. After receiving a packet from a sensing node, the storage node will extract each encrypted data item (say, $E_{k_\rho}(d_i^\rho)|L_i$ for $1 \le i \le \beta$ and $1 \le \rho \le \gamma$), which is composed of an n-bit encrypted data value (say, $E_{k_\rho}(d_i^\rho)$) and its least significant bit of neighborhood difference (say, L_i), by decomposing the ciphertext in the packet via Eq. 3. Since each data item has $n+1$ bits, it is easy for the storage node to retrieve each data item by slicing the received ciphertext every $n+1$ bits, where the first n-bit is the encrypted data value $E_{k_\rho}(d_i^\rho)$ and the last bit is L_i.

For the process of verifying rights, when querists want to query the storage node for desired data, the query packet $\{id, t, Q, \sigma\}$ will be sent. Upon the storage node receiving a query packet issued by a querist, Algorithm 3 will be used to verify the rights of the querist. If the verification is successful, the storage node will give a response to the querist; otherwise, the request is rejected.

After verifying the querist's rights successfully, the storage node will retrieve encrypted sensitive data in response to the querist's query. The storage node searches for the requested data stored in its database over the encryption domain. Specifically, the encrypted data stored in the storage node's database intersected to the query range is the QR for the querist, such as $E_{k_2}(4)$ and $E_{k_4}(7)$, as showed in the storage node of Fig. 3. More specifically, all the retrieved data associated with their corresponding $L_i's$, coupled with an encrypted BM (*e.g.*, $E_{k_b}(618)$ on the left-hand side of Fig. 3), are packaged together in a packet as the QR. On the other hand, if no matched data can be found, the storage node only sends an encrypted BM as the QR for the sake of checking whether the storage node cheats the querist by sending a fake QR. Eventually, the storage node responds the QR to the querist.

6. Data privacy analysis

In the proposed scheme, as described in Section 5, sensitive data are encrypted before being transmitted to and stored on the storage nodes. Depending on the sophisticated encryption primitives supplied in this paper, through cracking the cryptographic ciphers, attackers are computationally difficult to deduce the actual content of sensitive data. As a result, the security of our retrieval scheme is based on the security of the encryption scheme, named OEM in this paper, for sensitive data. Aiming to the security of OEM, we analyze the potential information leakage under the renowned attack model, called Chosen-plaintext attack (CPA). Recall that, as described in Section 3.2, the storage nodes are supposed to be un-trusted. In order to confirm the security strength of our method, the security definition and proofs are examined under the CPA model in this section.

Security Definition. Let $\mathcal{A}(A_1, A_2)$ be a polynomial time attacker. Given a security parameter $k \in \mathbb{N}$, a symmetric encryption scheme (*Gen, Enc, Dec*) is CPA secure if and only if the advantage

of the following experiments is negligible.

Experiment :

$$\mathcal{K} \leftarrow Gen(1^k)$$
$$(d_0, d_1, s) \leftarrow A_1(1^k)$$
$$b \overset{\$}{\leftarrow} \{0, 1\}$$
$$C_b \leftarrow Enc_k(d_b)$$
$$b' \leftarrow A_2(d_0, d_1, s, c_b)$$
$$Return \quad b',$$

where s is a string that captures A_1's state. The advantage $\mathcal{A}dv$ is defined as:

$$\mathcal{A}dv = |Pr(b' = b) - \frac{1}{2}|, \tag{3}$$

where the probability is taken over the choice of *b, Gen*, and *Enc*.

Theorem 1. *If the block cipher encryption is CPA-secure, then OEM is CPA-secure.*

proof Let OEM=(*Gen, Enc, Dec*). A challenger can use *Gen* to get a key $\mathcal{K} \leftarrow Gen(1^k)$, where $\mathcal{K} = (k_1, \ldots, k_\gamma)$. When A_1 generates two plaintexts d_0 and d_1 and sends these two plaintexts to the challenger, the challenger flips a coin to get b and encrypt d_b as follows:

1. Find the appropriate $k_b \in (k_1, \ldots, k_\gamma)$
2. Input an IV and the k_b to the block cipher encryption scheme, and then output a keystream KS_b
3. Calculate $KS_b \bigoplus d_b$
4. Permute $KS_b \bigoplus d_b$ to get a ciphertext c_b
5. Output c_b to the attacker $\mathcal{A}$.

Even $\mathcal{A}$ gets $KS_b \bigoplus d_b$, he/she cannot verify that b is 0 or 1. That is because KS_b is generated by a sufficient secure block cipher encryption scheme and $\mathcal{A}$ cannot get input parameters, $\mathcal{K}$ and IV. In addition, due to pseudorandom permutation, there are $O(n!)$ possibilities for c_b, which is typically underdetermined such that knowing $KS_b \bigoplus d_b$ will not help the attacker uniquely determine c_b. As a result, OEM is CPA-secure. It should be noted that ensuring OEM against the CPA model implies that it also can resist both ciphertext-only attack (COA) and known-plaintext attack (KPA).

7. System analysis

To formalize and evaluate our system, we conduct analysis of integrity, energy consumption, and anonymity of queries for m querists as follows.

(1) Integrity Analysis: Let the query range be $[e_{n_\alpha}, e_{n_\beta}]$, let the correct query result be $S^* = \{e_{n_i^*}|L_{n_i^*}, \ldots, e_{n_j^*}|L_{n_j^*}, \ldots, e_{n_k^*}|L_{n_k^*}\}$, and let the query result supplied from a storage node be $S = \{e_{n_i}|L_{n_i}, \ldots, e_{n_j}|L_{n_j}, \ldots, e_{n_k}|L_{n_k}\}$, where $\alpha < i < j < k < \beta$. The querist is able to detect the storage node's misbehaviors, including insertion of fake data items, deletion and tampering of qualified data items, if he/she follows a sequence of rules below.

- If there exists $n_i^* < n_j^* < n_k^*$ such that $e_{n_j^*} \notin S$ (which means a qualified data value is omitted or tampered) or if there exists $n_i < n_j < n_k$ such that $e_{n_j} \notin S^*$ (which signifies a fake data value is inserted), the querist can detect this mistake of omitting or tampering qualified data value. On the basis of X2L and BM, we can find out the incompleteness of query result by Case 1 of querist behavior, described in Section 5.3.
- If $n_i < n_i^*$ or $n_k > n_k^*$, the querist can also detect the error value n_i or n_k according to Cases 2 and 3 of querist behavior, described in Section 5.3.
- If *QR* only contains a BM value, then the querist only needs to recover the BM table and observes the data locations to further verify whether data values corresponding to the query of the querist exist or not.

Depended on the above verifications, we can detect forged and incomplete query results with probability $1 - negl()$, where $negl()$ is a negligible function, when storage nodes misbehave.

(2) Energy Consumption: We formulate energy consumption of communication between each sensing node - storage node pair and between each storage node - querist pair in our method. We first assume that packet transmission between each sensing node - storage node pair is Poisson process. Let $\{X(t), t \geq 0\}$ be a counting process in that $X(t)$ represents the exactly number of received packets that occur by the time t and let p be the probability that a packet is received. It can be modeled as a Poisson process with the probability $P(X(t) = k)$, where exactly k packets are received for the outcome of δ packets that are transmitted, being defined as:

$$P(X(t) = k) = C(\delta, k)p^k(1-p)^{\delta-k}, k = 0, 1, 2, \ldots, \delta. \tag{4}$$

Let the Poisson process have rate $\lambda = \delta p$. Eq. (4) can be rewritten as:

$$P(X(t) = k) = (e^{-\lambda t} \cdot (\lambda t)^k)/k!. \tag{5}$$

When a storage node receives packets from a sensing node by the time t, the theoretical energy consumption for receiving total packets by the time t is defined as:

$$E_{SNtoSTN} = P(X(t) = k) \cdot k(\beta + (\omega \cdot I + \beta \cdot x)/8) \cdot E_{rec}, \tag{6}$$

where β denotes the number of data values per packet, x denotes the number of bits required for neighborhood difference, ω denotes the size of each interval, I denotes the number of intervals, and E_{rec} denotes the energy consumption of receiving one byte of data.

Apart from analyzing energy consumption of each sensing node - storage node pair, we discuss energy consumption in a storage node - querist pair. If a storage node receives a query from a querist, the energy consumption consists of reception, DoS attack filtering, and right verification. Here, we denote energy consumption of DoS attack filtering and the right verification as E_{filter} and E_{verify}, respectively. Then, we define the total energy consumption required for a storage node to receive a query from a querist as:

$$E_{QTtoSTN} = (\chi + b_\sigma) \cdot E_{rec} + r \cdot E_{filter} + (1 - r) \cdot (E_{filter} + E_{verify}), \tag{7}$$

where both χ and b_σ take in bytes to signify the sizes of an encrypted query message and a signature, respectively, and r represents the percentage of suffering DoS attacks. Specifically, if there are κ fake packets[2] out of N communications, we can get the percentage of suffering DoS attacks by $r = \kappa/N$. Note that the energy consumption formulations in Eqs. (6) and (7) do not count the number of bits in representing data, such as destination *id*, source *id*, and time slot, since the delivery of these data is necessary in other state-of-the-art methods.

(3) Anonymity of queries for m querists: The notion is that if (compromised) storage nodes cannot identify which query belongs to which querist, they are hard to mount choice attacks to the specific querist. For being unidentifiable within a set of querists and quantifying the anonymity of queries, we first introduce a mathematical model [32] for analyzing the entropy of queries. Assume that there exist m querists who will issue a total of L queries to a storage node during a period of time, in which the ith querist will

[2] Fake packets are caused by attackers launching DoS attacks.

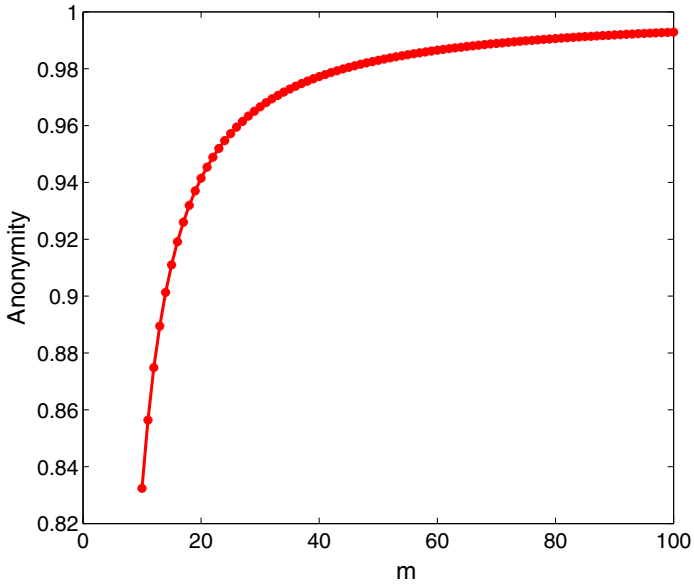
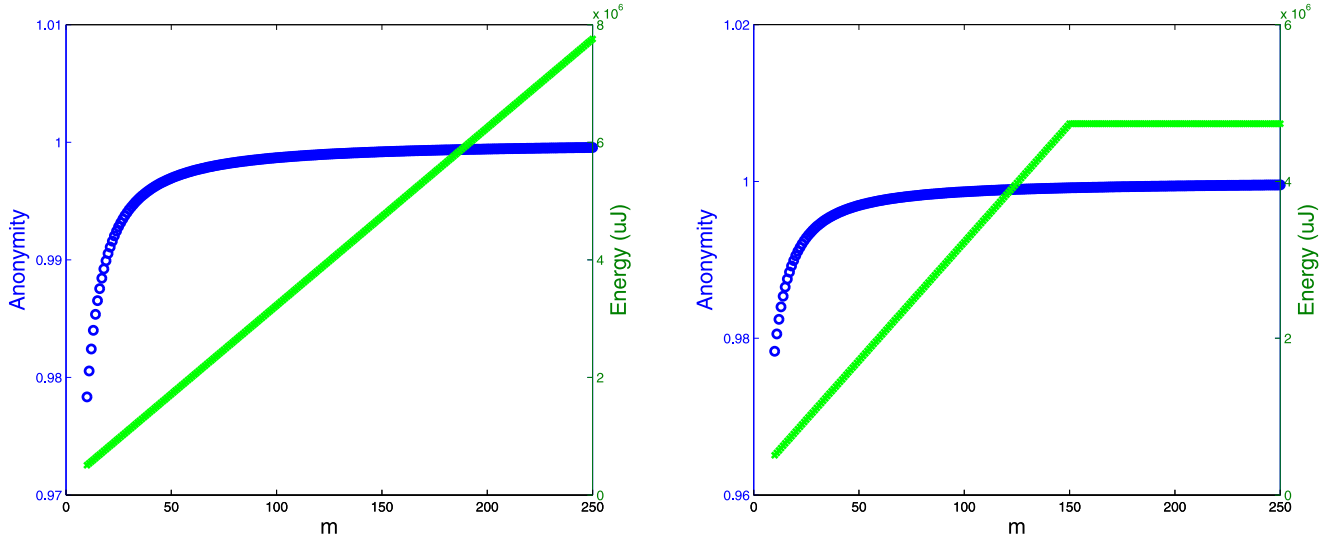


Fig. 5. Anonymity vs. m.



(a) $0 \leq m \leq 250$ and $r = 0$ (b) $0 \leq m \leq 250$, $r = 0$, and $RD_{max} = 150$

Fig. 6. Relationship between anonymity and energy consumption.

issue n_i $(i = 1, 2, \ldots, m)$ queries. The entropy of the set of probabilities $n_1/L, \ldots, n_m/L$ under m querists is defined as:

$$H(m) = -\sum_{i \in m} (n_i/Ł) \cdot log_2(n_i/Ł). \tag{8}$$

Therefore, to estimate the anonymity for m querists, which is considered as ratio degree ($RD(m)$) of m querists in our paper, $RD(m)$ can be calculated as:

$$RD(m) = H(m)/H_{max}, \tag{9}$$

where $H_{max} = \arg\max_m H(m)$.

Fig. 5 demonstrates the anonymity with an increase in m. It should be noted that a higher value of $RD(m)$ implies that querists' identities are more random. Nevertheless, this would cause higher energy consumption while a larger m is used, as shown in Fig. 6(a).

Thus, one must balance between energy consumption and anonymity. First, a formula stating the relationship between energy consumption and m is defined as:

$$E(m) = (\chi + \sigma) \cdot E_{rec} + r \cdot E_{filter} + (1 - r) \cdot (E_{filter} + RD_{max} \cdot E_{verify}), \tag{10}$$

where $RD_{max} = \arg\max_m \{RD(m)\}$ and $RD(m)$ is defined in Eq. (9). The relationship between anonymity and energy consumption, shown in Fig. 6(b), can be calculated via Eq. (10) at $0 \leq m \leq 250$. From Fig. 6(b), we can find that the anonymity can achieve the highest value at $m \geq 150$. Thus, RD_{max} can be defined as 150 for stationary energy consumption. In this way, a querist, who wants to query a storage node, just randomly chooses 149 sub-group querists from 249 group querists plus himself/herself to form a sub-group. Then, the sub-group with 150 group querists in total will be exploited by the querist to sign messages for hiding his/her identity.

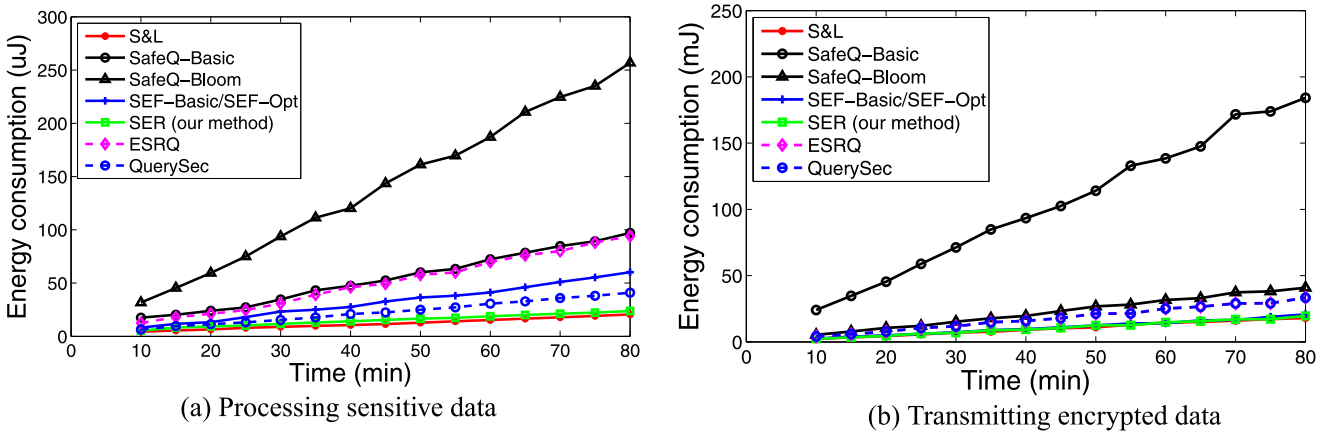


Fig. 7. Average energy consumption on a sensing node during a period of 80 minutes.

8. Performance evaluation

We present a publicly available implementation of SER for the TelosB platform in Section 8.1 and describe TOSSIM-based simulations in Section 8.2.

8.1. Experiments

Aiming to privacy preservation, S&L [4], SafeQ-Basic/SafeQ-Bloom [5], SEF [13], QuerySec [6], and ESRQ [10] were taken for comparison in four aspects: (1) energy consumption of processing sensitive data on a sensing node; (2) energy consumption of transmitting encrypted sensitive data from a sensing node to a storage node; (3) storage overhead on a storage node; and (4) energy consumption of a storage node in sending the query result and verification information[3] to a querist. Note that, for fair comparison, we do not consider the effects of access control while comparing the effects of privacy-preserving. We do not conduct comparison with the schemes in [11,12] because they may incur a serious security problem in that a compromised sensing node can easily send false bit maps to breach the integrity verification of the network. We also do not compare our method with SMQ [16] due to its vulnerable verification protocol and severe security breach in that the bucket index used in SMQ leaks the possible range of sensitive data for each sensor reading to attackers. In addition, CSRQ [8] is a variation of S&L, but it did not address the issue of false positives. As a result, we just compare our method with S&L, instead of CSRQ. As for anonymous access control, we examine the verification cost and execution time in both *Priccess* [9] and SER. Moreover, DoS attacks were also examined.

In this subsection, we present a publicly available implementation of SER for the TelosB platform in Section 8.1.1 and describe performance comparisons in Section 8.1.2.

8.1.1. Hardware module and experimental setup

TelosB is composed of an MSP430 micro controller, an 802.15.4 TI wireless transceiver, 10 KB RAM, 48 KB ROM, and 1024 KB flash memory. The platform fully supports TinyOS 1.X/2.X and implements a network stack. To implement security functionalities of state-of-the-art methods, we employed SHA-1 as a hash function in SafeQ-Basic/SafeQ-Bloom and SEF, applied the DES encryption algorithm in S&L and SafeQ-Basic/SafeQ-Bloom, applied the OPSE function proposed in [33] as the underlying OPSE function in SEF, and applied ECC for *Priccess*. For our method, we construct a provable secure mechanism, OEM, with 128-bit keystream, and an efficient authentication scheme, RFV. As for the network settings, the sensor network consists of 12 TelosB motes as sensing nodes, 9 TelosB motes as storage nodes, and one laptop as querists, where the sensing nodes communicate with the storage nodes in one-hop distance while they collect environmental information (*e.g.*, temperature, frequency of vibration, humidity, and etc). According to the experiments of [5,6,13], we adopted the time slots ranging from 10 to 80 minutes and, for each time slot, querists randomly generated range queries in the possible data domain.

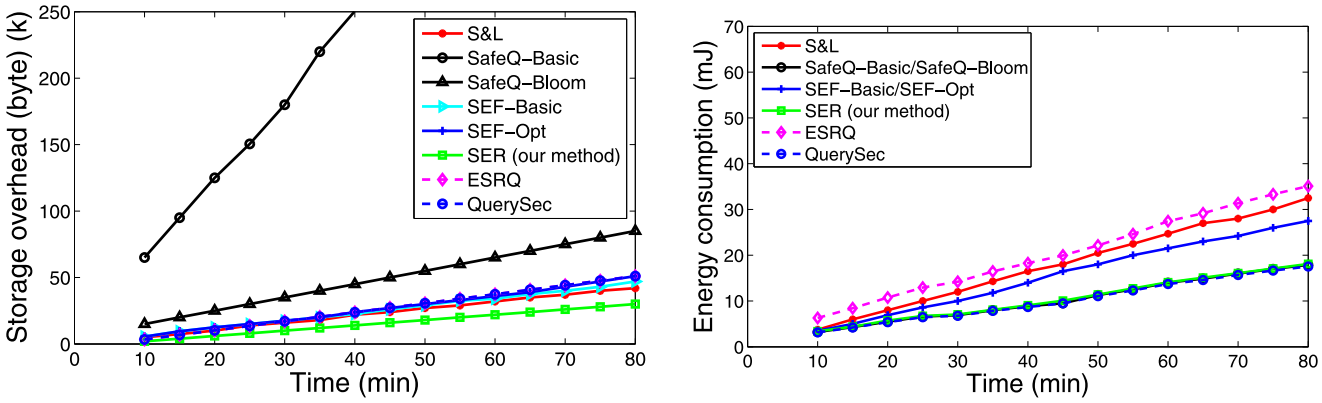
We implemented our method on TelosB in nesC, the programming language used for TinyOS. The ROM needed for sensing node and storage node was 15.4 KB and 22.8 KB, respectively. Furthermore, the RAM needed for sensing node and storage node was 2.8 KB and 1.6 KB, respectively. In addition, the laptop was connected with a TelosB and ran SER in C code. After generating a query packet, the laptop would send a query packet to a storage node through the TelosB. The laptop was used to simulate several querists' behaviors.

8.1.2. Experimental results

In measurement procedures, we used an Agilent 34411A power meter to measure the average current (under the assumption that the voltage supply is constant) drawn by the evaluation nodes (*i.e.*, sensing nodes or storage nodes). Subsequently, the measurements were displayed and recorded on a PC using the NI LabVIEW SignalExpress graphic user interface (GUI) software to communicate with the power meter.

Energy consumption of processing sensitive data. The energy consumption of processing sensitive data includes that of encrypting sensed data, building BM values and encapsulating data into a packet. In fact, data encryption accounts for the majority ratio of energy consumption. We have the following observations of the results shown in Fig. 7(a). (1) The energy consumption of SER is approximately equal to that of S&L even though SER has to perform X2L and generate a BM for incomplete verification. (2) Compared with SER, SEF-basic/SEF-opt consumes about 2.5 times more energy, SafeQ-Basic consumes about 4 times more energy, and SafeQ-Bloom consumes about 10.8 times more energy. (3) Compared with SER, ESRQ and QuerySec cost towards 3.8 times and 1.5 times more energy, respectively. This is because SEF-Basic/SEF-Opt has to construct the AI tree that needs to perform many hash operations and SafeQ-Basic/ SafeQ-Bloom adopts the prefix membership verification strategy, which still requires a large number of hash operations. Note also that, because SafeQ-Bloom has to perform a lot of additional hash operations to obtain the Bloom filter, it con-

[3] In our method, verification information means the least significant bit of neighborhood difference and the BM value associated with each plaintext.



(a) Average storage overhead. (b) Average communication overhead.

Fig. 8. A storage node overhead during a period of 80 minutes.

sumes more energy than SafeQ-Basic. In ESRQ, it is of necessity to generate a number of so-called special codes in accordance with a selected number of independent and randomly distributed hash functions. Aiming to QuerySec, it produces a link watermarking for each data value and applies privacy preserving function to each data value, leading to remarkable energy consumption.

Energy consumption of transmitting encrypted data. Fig. 7(b) shows the energy consumption of transmitting encrypted data from a sensing node to a storage node. We assumed that there were 10 data values sensed by a sensing node. As shown in Fig. 7(b), the energy consumption of SER was approximately equivalent to S&L and SEF, when there are 10 128-bit encrypted sensitive data. Nevertheless, since SafeQ needs to perform many operations of hashing and appends this hash value to the transmitted packet, it may cause more energy consumption when transmitting the packet from a sensing node to a storage node. Fig. 7(b) demonstrates that, when compared with SER, SafeQ-Basic and SafeQ-Bloom, respectively, consume about 12 times more energy and 3 times more energy. Observing from the energy consumption of ESRQ and QuerySec, they nearly cost the same energy in transmitting messages to a storage node because both of them have the same packet format. But they still consume 1.7 times more energy than our method.

Storage overhead on a storage node. The results regarding storage overhead, including encrypted data values and verification information, on a storage node are shown in Fig. 8(a). Our method is much better than the state-of-the-art methods. More specifically, the encoding numbers and buckets used in S&L require slightly more storage consumption than SER; SEF-Basic/SEF-Opt uses HMAC of the AI tree to provide verification information and consumes slightly more storage than SER; SafeQ-Basic consumes about 32.5 times more storage overhead; Safe-Bloom still consumes about 7.5 times more storage than SER; and ESRQ and QuerySec have the same storage overhead because of the same packet format transmitted from sensing nodes. Both of them cost about 1.7 times more storage space than SER.

Energy consumption of returning response. Fig. 8(b) depicts the energy consumption of a storage node in returning the query result and verification information to a querist. We can observe that SER consumes less energy than other methods. Specifically, SEF-Basic/SEF-Opt consumes about 1.6 times more energy than SER. Moreover, the false positive incurred by the bucking technique makes S&L consume about 1.8 times more energy than SER. In ESRQ, the query results and their corresponding so-called special codes are needed to be returned to a querist. In contrast to ESRQ, SER is only of necessity to return the least significant bits and BM value to a querist. Observing from Fig. 8(b), ESRQ costs about 2 times more energy than SER in returning response to a querist. Moreover, SER consumes a bit more energy than QuerySec and SafeQ in returning response to a querist. QuerySec and SafeQ employ a link watermarking and neighborhood chain technologies, respectively, which can embed verification objects in each data value. Therefore, a querist can verify the integrity of query result without appending any additional information into transmitting packets.

Execution time and energy consumption of query generation. We examine the execution time and energy consumption of generating query packets for a querist. The execution time and energy consumption for a querist's query generation under various numbers of group querists (m) and storage nodes (c) are depicted in Fig. 9 (a) and (b), respectively. The execution time and energy consumption are evaluated while the querist uses the RFV to sign various messages for the storage nodes existing in the network. Here, the querist's query generation is run on a laptop with 1.8 GHz CPU. In the case of $m = 30$ and $c = 9$, the execution time and energy consumption are about 1440 ms and 6567 uJ, respectively, under the average of performing the same experiment two hundreds.

Execution time of query verification. Fig. 10 shows the verification execution time of our method compared with that of *Priccess* on TelosB. With an increase in m, the execution time is also increased but it is independent of the scale of a WSN. We find that, even though our method needs to spend time filtering DoS attacks, the execution time of our method still approximates that of *Priccess*. In the case of $m = 30$ and $c = 9$, the execution time of our method is about 1.234 s and that of *Priccess* is about 1.224 s.

Verification effects for a storage node. We estimated the energy consumption of right verification and show the energy consumption of verifying a querist's rights for a storage node in Table 2. Although our method needs slightly more cost than *Priccess*, our method will take less energy consumption and execution time, as shown in Fig. 11(a) and (b), respectively, under the case of suffering DoS attacks.

For evaluating the verification cost under the case of suffering DoS attacks, we evaluate the energy consumption via Eq. (7) and show the numeric results in Fig. 11(a). Since the amount of current draw on normal operations for TelosB is 21.8 mA, we calculate the communication cost as follows. Given a 250 kbps radio transmission rate and 29-byte packet, it takes $29 \cdot 8(\text{bits})/250(\text{kbps}) = 0.928(\text{ms})$ for a sensor node to receive a packet. Then, the energy cost for receiving one packet (E_{rec}) is $3.0(\text{v}) \cdot 21.8(\text{mA}) \cdot 0.928(\text{ms}) = 60.6912(\mu\text{J})$. According to Fig. 11(a), it costs $2.9406 \cdot 10^5(\mu\text{J})$ in the case of 20 fake packets out of 50 communications at $m = 10$. During the authentication, it takes one packet for the querist to send

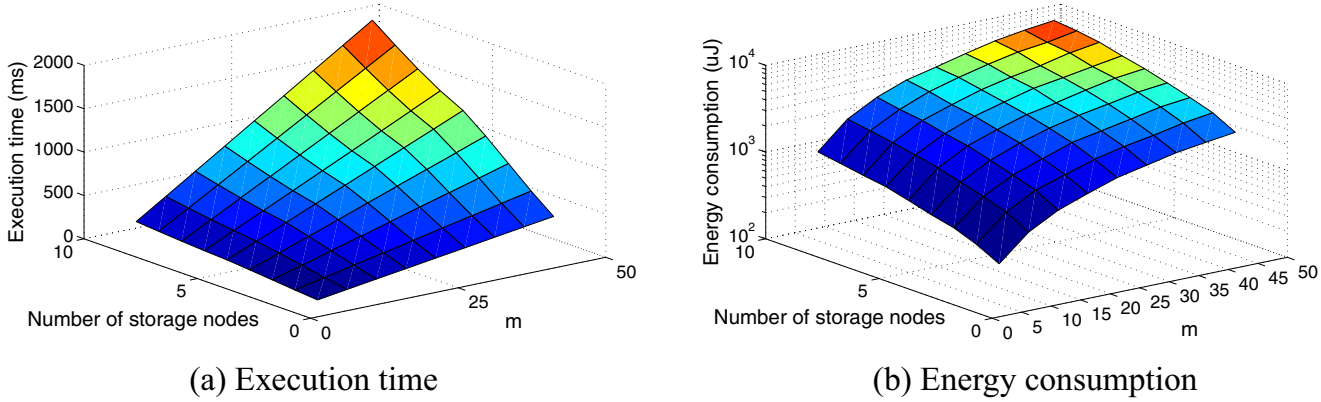


Fig. 9. Execution time and energy consumption for a querist's query generation under various numbers of group querists (m) and storage nodes.

Table 2
Verification cost for a storage node.

Group querists (m)	10	15	20	25	30	35	40	45	50
Node verification cost (*Priccess*) (J)	0.400	0.551	0.711	0.864	1.001	1.177	1.332	1.511	1.622
Node verification cost (SER) (J)	0.457	0.603	0.765	0.921	1.057	1.231	1.388	1.567	1.676

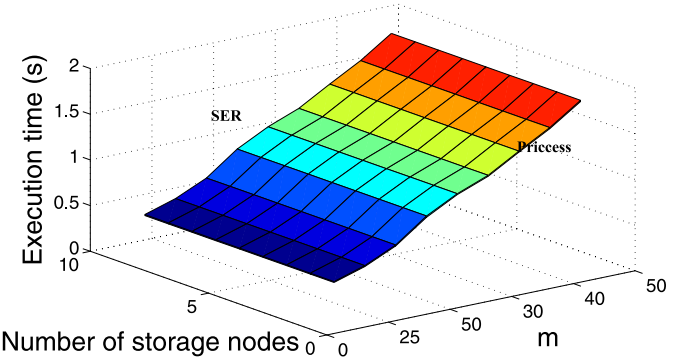


Fig. 10. Execution time for verifying a querist's query under various numbers of group querists (m) and storage nodes.

a query to the storage node in our method. Our experiments show that SER takes less energy consumption than *Priccess* under various numbers of fake packets and communications.

Fig. 11(b) shows the verification time comparison between *Priccess* and SER when DoS attacks were considered. Observably, our method can filter the attacks while taking less execution time than *Priccess*.

8.2. Simulations

We also implemented S&L, SafeQ, SEF, ESRQ, QuerySec, and SER schemes using TOSSIM [34] to evaluate communication cost in a large scale network. The network was set with the number of sensing nodes ranging from 200 to 600 in a $200m \times 200m$ rectangular region. The real world data set was collected in a physical environment, which consists of the temperature, humidity, and frequency of vibration. The obtained simulation results show the impact of network size on communication for sensing nodes and storage nodes. As we can observe from Fig. 12(a), more data are transmitted from sensing nodes to storage nodes while network size grows. It is evident that SafeQ-Basic/SafeQ-Bloom consumes more communication cost than other schemes due to its prefix family members. In ESRQ and QuerySec, they need to attach a number of verifiable objects to each sensed data. Therefore, these two

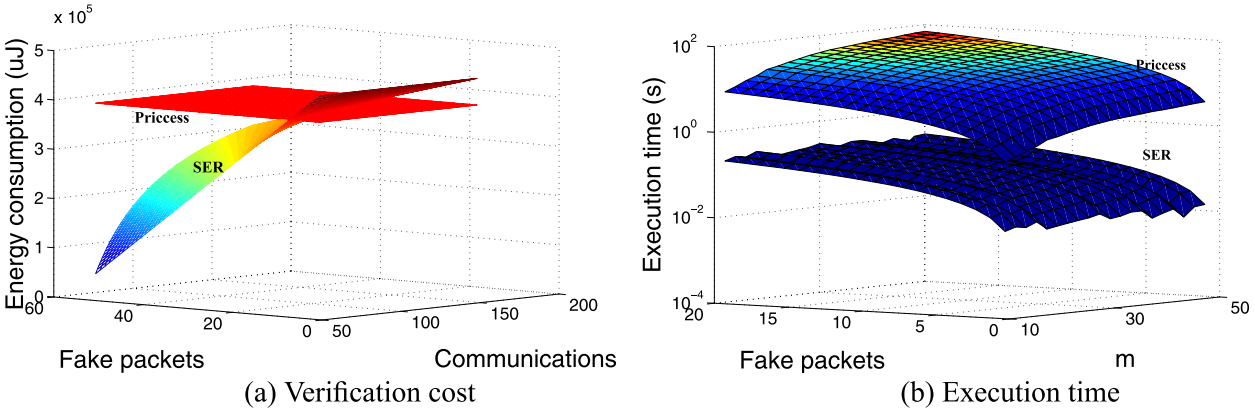


Fig. 11. A comparison of verification cost and execution time between our method SER and *Priccess* under DoS attacks. X-axis in Figs. (a) and (b), respectively, denotes the number of packets communicated in the air and group querists (m), Y-axis in both Figs. (a) and (b) denotes the number of fake packets caused by attackers launching DoS attacks, and Z-axis in Figs. (a) and (b), respectively, denotes energy consumption and execution time for verifying packets by a storage node under various numbers of fake packets.

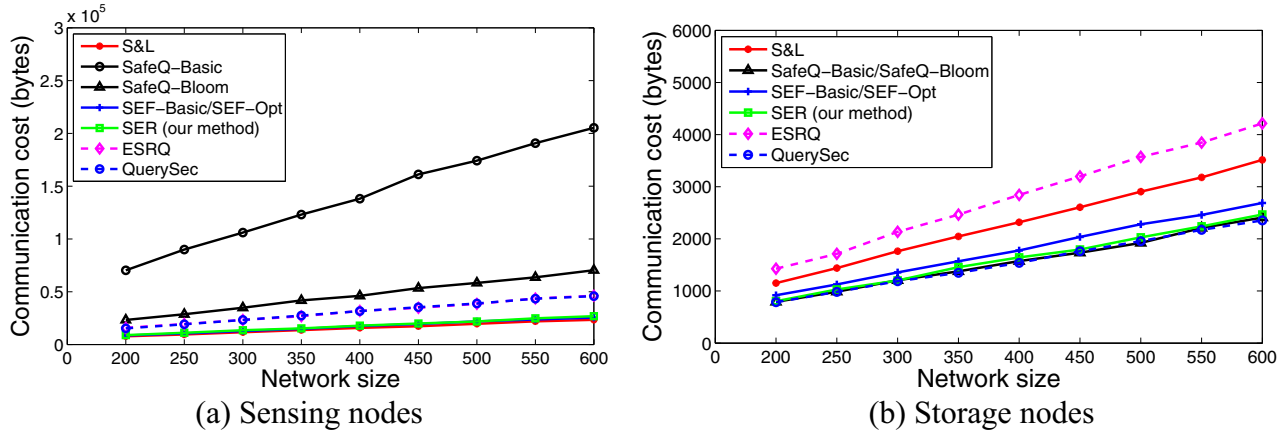


Fig. 12. Impact of network size on communication cost.

schemes consume more communication cost than SER, which only generates a bit of LSBs and a BM table as the attached code. In addition, SEF-Basic/SEF-Opt and S&L produce as less verifiable objects as SER and, thus, SEF-Basic/SEF-Opt, S&L, and SER consume the similar communication cost.

As shown in Fig. 12(b), ESRQ consumes the most communication cost because there is a great deal of so-called BFcodes attached to the query result. S&L costs more than SER due to the false positives and encoding numbers. Moreover, SEF-Basic/SEF-Opt also needs more cost than SER due to a number of verifiable objects is still required. Since QuerySec and SafeQ employ a link watermarking and neighborhood chain technologies, respectively, which can embed verification objects in each data value, querists can verify the integrity of query results without needing to append any additional information into transmitting packets while SER is still of necessity to attach a bit of attached codes to query results.

As shown in Fig. 12(a) and (b) that demonstrate the impact of network size on communication cost, one can see that storage nodes consumes less energy than sensing nodes because storage nodes only return satisfactory data to querists. Although QuerySec and SafeQ take less communication cost than SER for storage nodes, they consume much more communication cost than SER for sensing nodes. In view of the whole network, SER still exhibits the best overall performance among the methods adopted for comparisons.

9. Conclusions

This paper focuses on data privacy and integrity preservation for anonymous range query processing in wireless sensor networks. We present a provably secure OEM for effectively retrieving stored sensitive data over the encryption domain while constructing the efficient data structures X2L and Bit-Map table to guarantee the integrity of query results. More importantly, we develop a RFV scheme, which has capabilities to provide AAC for querists and resist DoS attacks for storage nodes. RFV can facilitate significantly reducing verification execution time and cost under DoS attacks while preserving anonymity of queries for all querists. Extensive experiments and simulations show that our method is more flexible and efficient than state-of-the-art methods even though storage nodes encounter choice attacks and DoS attacks.

Acknowledgment

This work was supported by Ministry of Science and Technology under Grants MOST 98-2221-E-001-004-MY3 and MOST 102-2221-E-001-002-MY2.

References

[1] Taking the pulse of the planet: Epa's remote sensing information gateway, http://www.epa.gov/geoss/.
[2] Nopp, http://www.nopp.org/.
[3] B. Carbunar, Y. Yu, L. Shi, M. Pearce, V. Vasudevan, Query privacy in wireless sensor networks, ACM Trans. Sens. Netw. (TOSN) 6 (2) (2010) 1–34.
[4] B. Sheng, Q. Li, Verifiable privacy-preserving range query in two-tiered sensor networks, in: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), 2008, pp. 457–465.
[5] F. Chen, A.X. Liu, Safeq: secure and efficient query processing in sensor networks, in: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), 2010, pp. 1–9.
[6] Y. Yi, R. Li, F. Chen, A.X. Liu, Y. Lin, A digital watermarking approach to secure and precise range query processing in sensor networks, in: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), 2013, pp. 1950–1958.
[7] L. Dong, X. Chen, J. Zhu, H. Chen, K. Wang, C. Li, A secure collusion-aware and probability-aware range query processing in tiered sensor networks, in: Proceedings of the IEEE 34th Symposium on Reliable Distributed Systems (SRDS), 2015, pp. 110–119.
[8] H. Dai, Q. Ye, G. Yang, J. Xu, R. He, Csrq: communication-efficient secure range queries in two-tiered sensor networks, J.Sensors 16 (2) (2016) 259–275.
[9] H. Daojing, J. Bu, S. Zhu, S. Chan, C. Chen, Distributed access control with privacy support in wireless sensor networks, IEEE Trans. Wireless Commun. 10 (10) (2011) 3472–3481.
[10] X. Zhang, L. Dong, H. Peng, H. Chen, D. Li, C. Li, Achieving efficient and secure range query in two-tiered wireless sensor networks, in: Proceedings of the IEEE/ACM International Symposium on Quality of Service (IWQoS), 2014, pp. 380–388.
[11] J. Shi, R. Zhang, Y. Zhang, Secure range queries in tiered sensor networks, in: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), 2009, pp. 945–953.
[12] R. Zhang, J. Shi, Y. Zhang, Secure multidimensional range queries in sensor networks, in: Proceedings of the Tenth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 2009, pp. 197–206.
[13] B. Jiajun, Y. Mingjian, H. Daojing, X. Feng, C. Chun, Sef: a secure, efficient, and flexible range query scheme in two-tiered sensor networks, Int. J. Distrib. Sens. Netw. 2011 (2011) 1–12.
[14] L. Dong, J. Zhu, X. Zhang, H. Chen, C. Li, H. Sun, Semr: secure and efficient multi-dimensional range query processing in two-tiered wireless sensor networks, in: Springer Web-Age Information Management, 2015, pp. 520–524.
[15] H. Hacigumus, B.R. Iyer, C. Li, S. Mehrotra, Executing sql over encrypted data in the database service provider model, in: Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data (SIGMOD), 2002, pp. 216–227.
[16] C.-M. Yu, Y.-T. Tsou, C.-S. Lu, S.-Y. Kuo, Practical and secure multidimensional query framework in tiered sensor networks, IEEE Trans. Inf. Forensics Secur. 6 (2) (2011) 241–255.
[17] Y. Lee, Secure ordered bucketization, IEEE Trans. Dependable Secure Comput. 11 (3) (2014) 292–303.
[18] X. Zhang, L. Dong, H. Peng, H. Chen, S. Zhao, C. Li, Collusion-aware privacy-preserving range query in tiered wireless sensor networks, J. Sens. 14 (12) (2014) 23905–23932.
[19] H. Wang, Q. Li, Distributed user access control in sensor networks, in: IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS), 2006, pp. 305–320.
[20] R. Zhang, Y. Zhang, K. Ren, Distributed privacy-preserving access control in sensor networks, IEEE Trans. Parallel Distrib. Syst. 23 (8) (2012) 1427–1438.
[21] R. Pietro, L. Mancini, C. Soriente, A. Spognardi, Data security in unattended wireless sensor networks, IEEE Trans. Comput. 58 (11) (2009) 1500–1511.

[22] H. Wang, B. Sheng, Q. Li, Elliptic curve cryptography based access control in sensor networks, Int. J. Secur. Netw. 1 (3/4) (2006) 127–137.
[23] C.M. Yu, Y.T. Tsou, C.S. Lu, S.Y. Kuo, Constrained function-based message authentication for sensor networks, IEEE Trans. Inf. Forensic Secur. 6 (2) (2011) 407–425.
[24] A. Liu, P. Ning, Tinyecc: a configurable library for elliptic curve cryptograph in wireless sensor networks, Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN) (2008) 245–256.
[25] R. Curtmola, J. Garay, S. Kamara, R. Ostrovsky, Searchable symmetric encryption: improved definitions and efficient constructions, in: Proceedings of the 13th ACM Conference on Computer and Communications Security, 2006, pp. 79–88.
[26] E. Stefanov, C. Papamanthou, E. Shi, Practical dynamic searchable encryption with small leakage, in: Proceeding Network Distributed System Security Symposium, 2014, pp. 832–848.
[27] Y.-T. Tsou, C.-S. Lu, S.-Y. Kuo, Motesec-aware: a practical secure mechanism for wireless sensor networks, IEEE Trans. Wireless Commun. 12 (6) (2013) 2817–2829.
[28] J. Katz, Y. Lindell, Introduction to Modern Cryptography: Principles and protocols, in: Cryptography and Network Security Series, Hapman & Hall/CRC, 2007, pp. 389–420.
[29] S. Yoo, K. Park, J. Kim, A security-performance-balanced user authentication scheme for wireless sensor networks, Int. J.Distrib. Sens. Netw. 8 (3) (2012) 1–11.
[30] D. He, N. Kumar, M. Khan, J. Lee, Anonymous two-factor authentication for consumer roaming service in global mobility networks, IEEE Trans. Consum. Electron. 59 (4) (2013) 811–817.
[31] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, D. Culler, Spins: security protocols for sensor networks, ACM Wireless Netw. 8 (5) (2002). 521–234
[32] C.-E. Shannon, A mathematical theory of communication, Bell Syst. Tech. J. (1984) 623–656.
[33] A. Boldyreva, N. Chenette, Y. Lee, A. O'Neill, Order-preserving symmetric encryption, in: Proceedings of the 28th Annual International Conference on Advances in Cryptology, 2009, pp. 224–241.
[34] PowerTOSSIM, Powertossim: Efficient power simulation for tinyos applications, http://www.eecs.harvard.edu/shnayder/ptossim/(2004).