Journal of King Saud University – Computer and Information Sciences (2016) 28, 262–275

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

A secure data routing schema for WSN using Elliptic Curve Cryptography and homomorphic encryption

Mohamed Elhoseny [a,c,*], Hamdy Elminir [b], Alaa Riad [c], Xiaohui Yuan [a]

[a] Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA
[b] Department of Electrical Engineering, Kafr El-Sheikh University, Kafr El-Sheikh, Egypt
[c] Department of Information Systems, Mansoura University, Mansoura, Egypt

Received 2 August 2015; revised 17 October 2015; accepted 5 November 2015
Available online 7 December 2015

KEYWORDS

Wireless sensor networks;
Data encryption;
Secure clustering;
Energy consumption

Abstract Despite the great efforts to secure wireless sensor network (WSN), the dynamic nature and the limited resources of sensor nodes make searching for a secure and optimal network structure an open challenge. In this paper, we propose a novel encryption schema based on Elliptic Curve Cryptography (ECC) and homomorphic encryption to secure data transmission in WSN. The proposed encryption schema is built upon GASONeC algorithm (Elhoseny et al., 2014) that uses genetic algorithm to build the optimum network structure in the form of clusters. ECC is used to exchange public and private keys due to its ability to provide high security with small key size. The proposed encryption key is 176-bit and is produced by combining the ECC key, node identification number, and distance to its cluster head (CH). To reduce energy consumption of CH, homomorphic encryption is used to allow CH to aggregate the encrypted data without having to decrypt them. We demonstrated that the proposed method is capable to work with different sensing environments that need to capture text data as well as images. Compared with the state-of-the-art methods, our experimental results demonstrated that our proposed method greatly improve the network performance in terms of lifetime, communication overhead, memory requirements, and energy consumption.

* Corresponding author: Department of Information Systems, Mansoura University, Mansoura, Egypt.
E-mail addresses: Mohamed.elhoseny@unt.edu, mohamed_elhoseny@mans.edu.eg (M. Elhoseny), hamdy_elminir@eng.kfs.edu.eg (H. Elminir), amriad2000@mans.edu.eg (A. Riad), xiaohui.yuan@unt.edu (X. Yuan).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

1. Introduction

Forming sensor clusters is an effective way to improve scalability and longevity of wireless sensor network (WSN) (Li et al., 2013). However, security is a challenging issue in cluster-based WSNs, since sensors are usually deployed in unattended environments. Moreover, limited memory, processing power, and communication range of sensor nodes (SNs) make traditional cryptographic schemes infeasible. In addition,

http://dx.doi.org/10.1016/j.jksuci.2015.11.001
1319-1578

cluster heads (CHs) are usually responsible for data aggregation and consume more energy than the member nodes, which cause early termination from energy exhaustion. Furthermore, data decryption and encryption to ensure secure transmission demand more computation and hence shorten the network life.

To address these challenges, many methods have been developed to ensure security in WSN such as dynamic key change, complexity, CH election criteria, etc. (Ganesh and Amutha, 2013; Huang et al., 2014). Xiao et al. proposed SLEACH (Xiao et al., 2005) to build a secure clustering model for WSN. It aims to prevent sinkhole, selective forwarding, and HELLO flooding attacks. However, conventional encryption-based methods in general and SLEACH in particular are constrained by the memory size and faces degradation of network performance and shortening of its lifetime. To overcome the complexity of conventional encryption schemes in terms of storage space in WSN, Elliptic Curve Cryptography is used for its ability to provide high security with short key size. In addition, the homomorphic encryption schema (Zhou et al., 2014) is used to allow CHs to aggregate encrypted messages without a decryption process so that it has a short aggregation delay. Moreover, it prevents the attackers from gaining knowledge of data or the encryption information even if a CH is compromised. TinyPEDS (Girao et al., 2007) was proposed as a method for secure data aggregation at the CHs using privacy homomorphic encryption. Yet, this method is vulnerable to node compromise attacks (Zhou et al., 2014). Similarly, SEDA-ECC (Zhou et al., 2014) was developed following the principles of homomorphic encryption achieving improved results. However, the limited memory size and greater demand of energy consumption remain challenges for bringing security to WSN.

In this paper, we propose an encryption schema based on ECC and homomorphic encryption to secure data transmission in WSN with dynamically clustered sensor nodes. With the goal of introducing minimum impact to the lifespan of the entire network, genetic algorithm is employed to search for the most suitable sensor nodes as the cluster heads to relay messages to the base station. ECC is used to generate public and private keys for sensor nodes. The encryption key at each sensor node is of 176 bits and is produced by combining the ECC key, identification number, and distance to its CH. To reduce energy consumption of cluster head and cluster head compromised attack, homomorphic encryption is used to allow each CH to aggregate the encrypted data of its member nodes without a decryption process. The proposed Encryption method is built upon GASONeC algorithm (Elhoseny et al., 2014) which uses Genetic Algorithm (GA) to determine the optimum network structure that reduces the energy exhaustion after each transmission round. GASONeC is briefly explained at Section 4

The remaining of this paper is organized as follows: Section 2 clarify the main objectives and the novel contributions of this work. Section 3 presents the related work of secure WSN that achieves extended network life and avoid attacks to WSN. Section 4 discusses our proposed method to secure data transmission in dynamic clustered sensors in WSN. Section 5 discusses our experimental results and the contribution. Section 6 concludes this paper with a summary.

2. Objectives and contributions

The limited resources of WSN enforce most of researchers to trade off between the network availability and data transmission security. Although many attempts were done to balance between these two goals, the dynamic nature of WSN and the limited resources of sensor nodes make searching for a secure and optimal network structure an open challenge. Based on that, this work has two main objectives. The first objective is to build a dynamic and a self-organizing WSN structure that maximizes the network longevity as long as possible by balancing the energy consumption between all nodes during the network lifetime. The second objective is to secure the data transmission process from sensor nodes to the base station taking into consideration the network lifetime.

The contributions of this work is three-fold. First, a dynamic and optimal network construct is obtained by GASONec using genetic Algorithm and a new cryptography scheme based on ECC and homomorphic encryption is applied for secure data transmitting from sensor nodes to the base station. The new scheme aims to achieve balanced energy consumption across all nodes to prevent the high energy consumption of the CH through data aggregation process. The main advantage of the proposed scheme is its ability to deal with limited sensor resources, i.e., memory size and processing power.

Second, the encryption method can be used for both text and image data encryption. This method aims to be applied with different sensing environments that need to capture text data as well as images.

Third, the data aggregation method enables the cluster head to aggregate the encrypted data without decrypting first and re-encrypting to avoid CH energy consumption. This process is executed using Homomorphic encryption and can resist the attack of cluster head capture.

3. Related work

SLEACH (Xiao et al., 2005) protocol is the first attempt to build a secure version of the widely used LEACH protocol. It is prevents sinkhole, selective forwarding and HELLO flooding attacks. SLEACH prevents an intruder node to send falsified data messages. But it ensures no guarantee to confidentiality and availability. Based on SLEACH, SecLEACH (Oliveira et al., 2007) was proposed as an improvement to introduce a symmetric key and one-way hash chain. The main aim of that is to provide different performance numbers on efficiency and security depending on its various parameter values. Although it provides authenticity, confidentiality, integrity and freshness for node-to-node communication, SecLEACH did not provide a solution for the compromised CH attack. This is because SecLEACH is vulnerable to key collision attacks and do not provide full connectivity. Furthermore, TinyPEDS (Girao et al., 2007) was proposed as a novel scheme for secure data aggregation at the CH based on privacy homomorphic encryption. However, this schema cannot resist node compromise attacks (Zhou et al., 2014).

SEDA-ECC (Zhou et al., 2014) was proposed as a Secure-Enhanced Data Aggregation based on Elliptic Curve Cryptography. Similar to TinyPEDS, the design of SEDA-ECC is

based on the principles of privacy homomorphic encryption. The security results of this method are better than TinyPEDS especially in node compromise attack. But, the required memory size and the energy consumption represent the main challenges. Another method based on the identity digital signature (IBS) scheme called SET-IBS (Huang et al., 2014) was proposed for secure and efficient data transmission for cluster based WSN. In SET-IBS, security relies on the hardness of the Diffie–Hellman problem in the pairing domain. It provides a good solution for both active and passive attacks but the network lifetime still represents the main problem. Due to large computational overhead of the asymmetric cryptography, BGN (Boneh et al., 2005) was constructed on a cyclic group of elliptic curve point. In (Mykletun et al., 2006), several public-key-based encryptions schemes, i.e., EC-OU, were proposed to achieve data concealment in WSNs. A Well-known encryption schema is the Advanced Encryption System (AES). This algorithm is based on block cipher encryption. It has a fixed block size of 128 bits and has a key size of 128, 192, or 256 bits. However, AES running on 10, 12, and 14 rounds for 128-, 192-, and 256-bit key respectively is still found vulnerable by the researchers (Bogdanov et al., 2011). In addition, AES is not suitable for WSNs as it needs more hardware resources (XiaoJun et al., 2012).

Another widely used block cipher schema in WSN is Skipjack (Eryilmaz et al., 2009). It uses an 80-bit key to encrypt or decrypt 64-bit data blocks. This makes it vulnerable to the key search attack due to the short key length (Biham et al., 2005). However, the strategy is the main challenge of SkipJack in WSNs. To avoid the short key problem of Skipjack, TWINE (Suzaki et al., 2013) uses an 80 bit or 128 bit key. The best known attacks against TWINE are two biclique attacks with a data requirement for the two attacks equal to 260 (Karakoca et al., 2013). Based on lightweight block cipher schema, LED (Isobe and Shibutani, 2012) was proposed to encrypt 64 bit blocks using either 64 bit or 128 bit key. To generate the new key in LED, the key is XORed at every four rounds. LED has two main limitations, it cannot prevent man-in-the middle attack and consumes more CPU cycles compared to conventional cryptographic schemes (Biswas and Muthukkumarasamy, 2014).

BCC (Liu and Tian, 2012) is a block cipher that avoids floating-point operations and multiplications in order to minimize the energy consumption. It is a flexible encryption method that supports different number of rounds and length of the encryption key. But, the authors failed to present any security and performance analysis for it. In addition, an encryption schema based on EEC and Chaotic Map was proposed in Biswas and Muthukkumarasamy (2014). The proposed cryptographic scheme employs elliptic curve points to verify the communicating nodes and as one of the chaotic map parameters to generate the pseudo-random bit sequence. This sequence is used in XOR, mutation and crossover operations in order to encrypt the data blocks. However, the memory size is the big challenge of this schema. SecDAO-LEACH (Saminathan and Karthik, 2013) was proposed to enhance the WSN performance in terms of security, reliability and fault-tolerance. This protocol is resistant to security attacks such as, replay attacks, node compromising attacks and impersonation attacks. In addition, it performs better in terms of energy consumption. However, it suffers from the limited resources of sensors specially the required memory size.

EECBKM (Lalitha and Umarani, 2012) is a cluster based technique for key management in which the clusters are formed in the network and the CHs are selected based on the energy cost, coverage and processing capacity. An EBS key set is assigned by the base station to every CH and cluster key to every cluster; this proposed technique reduces node-capture attacks and efficiently increases packet delivery ratio with reduced energy consumption. But the problem of this protocol is that it works well in the environment with low density sensors. In addition, it suffers many kinds of active attack. Another method is the SAC which is successful in preventing attacks caused by adversary like hello flooding and provides resilience to sensor nodes captured by adversary (Singh and Hussain, 2010). PIKE uses probabilistic techniques to establish pair wise keys between neighboring nodes in the network. However, in this approach, each node has to store a large number of keys.

Another secure clustering algorithm is SCMRP (Kumar and Jena, 2010) which is based on multi-path technique. SCMRP collects the benefits of both cluster based routing and multi-path routing. It provides security against various attacks like altering the routing information, selective forwarding attack, sinkhole attack, wormhole attack, Sybil attack, etc. In addition, it uses cryptography as a security mechanism to protect message after portioning it to packets. SCMRP consists of five phases: neighbor detection and topology construction, pairwise key distribution, cluster formation, data transmission, and re-clustering and rerouting. The base station collects all the neighbor list from sensor node and applies an algorithm called DFS for finding multiple paths. The BS generates the pairwise key and unicasts to all nodes. The CH selection is based on the remaining energy of the node.

SHEER (Ibriq and Mahgoub, 2006) aims to create a secure clustering schema with energy-efficient and secure communication on the network layer. SHEER uses the cryptography as the security mechanism. It proposed a schema for key distribution based on the Hierarchical Key Establishment System (HKES). SHEER proposed also a probabilistic transmission mechanism to reduce energy consumption and extend the network lifetime. This method works effectively against HELLO flood attack, sybil attack and sinkhole attack. Its main drawback is that it is not able to protect the network from selective forwarding attacks.

AKM (Kausar et al., 2008) is a cryptography-based method that provided security by using two kinds of keys: a pair-wise between the nodes inside the cluster, and a network key. This algorithm provides multiple levels of encryption that work well with secure cluster formation and avoid node captures. AKM provides confidentiality, continuous authentication of nodes in the network by periodically changing the network key. However, if the compromised node attached with the network before refreshing the current network key, all the network operations of can be monitored.

In IKDM (Cheng and Agrawal, 2007) each node has a unique identifier (ID) in the network. It uses Pairwise key a mechanism for cryptography. The node ID is assigned at the initialization phase of the network by an offline Key Distribution Server (KDS). Then every node creates a pair-wise key between them by exchanging their node IDs first. This method provides better network throughput and fixed key storage overhead and is suitable for large-scale WSNs. Therefore IKDM scheme is more energy-efficient due to the lower

communication overhead for sensor nodes during the pair-wise key establishment process. Also, it can achieve better network resilience against node capture attack.

4. Methodology

Fig. 1 shows the phases of our proposed work. The first phase is related to constructing the network structure that minimizes energy exhaustion using GASONeC algorithm. Then, the proposed encryption schema is applied to guarantee secure data routing from sensor nodes to the BS. These phases are discussed in detail in the following.

4.1. Dynamic cluster building using GASONeC

GASONeC (Elhoseny et al., 2014) was proposed as a Genetic Algorithm-based, clustering method that optimizes dynamic node clustering to extend the network lifetime. In GASONeC, the remaining energy, the expected energy expenditure, the distance to the base station, and the number of nodes in the vicinity are employed to search for an optimal, dynamic network structure. Balancing these factors is the key of organizing nodes into appropriate clusters and designating a surrogate node as cluster head. The factors are encoded into the fitness function of Genetic Algorithm (GA). In the optimization process, each GA chromosome represents a designation map of cluster heads. A gene in a chromosome specifies if the corresponding node serves as a cluster head. Given a cluster head, the node clusters are then formed following the nearest neighbor rule, and the fitness of a WSN structure prescribed by a chromosome is hence determined by the evaluation of all clusters. In each transmission round, network structure is updated dynamically to achieve network longevity. To ensure security in such a dynamically clustered sensor network, the security protocol must also take energy consumption into consideration.

Fig. 2 shows the main steps of GASONeC work. First, a binary chromosome is used to specify the cluster heads in the network, in which one represents a cluster head and zero represents a member node to a cluster. When a node becomes inactive, i.e., out of power, its corresponding gene value is set to -1, which exempts the node from further GA operations. An example for the binary chromosome is shown in Fig. 3 and the cluster heads are highlighted with filled circles. The dash arrows depict the cluster membership. The length of the chromosome is determined by the number of sensor nodes in the network field. To construct the network, distances between non-CH nodes and CH nodes are calculated. The sensor nodes are organized into clusters following the nearest neighbor rule. So, each chromosome is translated to a network structure. That is, a non-CH node s is assigned to a cluster if the CH of that cluster is closer to s than any other CHs in the network. The total distance to the CHs is hence minimized.

The third step is to use the fitness function to evaluate the proposed network structure. The optimization goal is to minimize the remaining energy variation among nodes. The fitness function includes a set of parameters as the remaining energy $\widetilde{E}$ and the expected energy expenditure ΔE. In addition, the distance to BS $\overline{D}$ and the number of neighbors N_e of a node i are included to diversify the cluster structure. So, the fitness function of a proposed network structure φ is defined as follows:

$$f(\varphi) = \frac{\widetilde{E}}{NE(0)} + \frac{E'}{\Delta E} + \frac{1}{\overline{D}} + N_e \tag{1}$$

where E' denotes the total energy cost if the messages are transmitted directed from the sensor nodes to the BS. The value of $\overline{D}$ is calculated as

$$\overline{D} = \sum_{i=1}^{|ch|} d_{chi,BS} \tag{2}$$

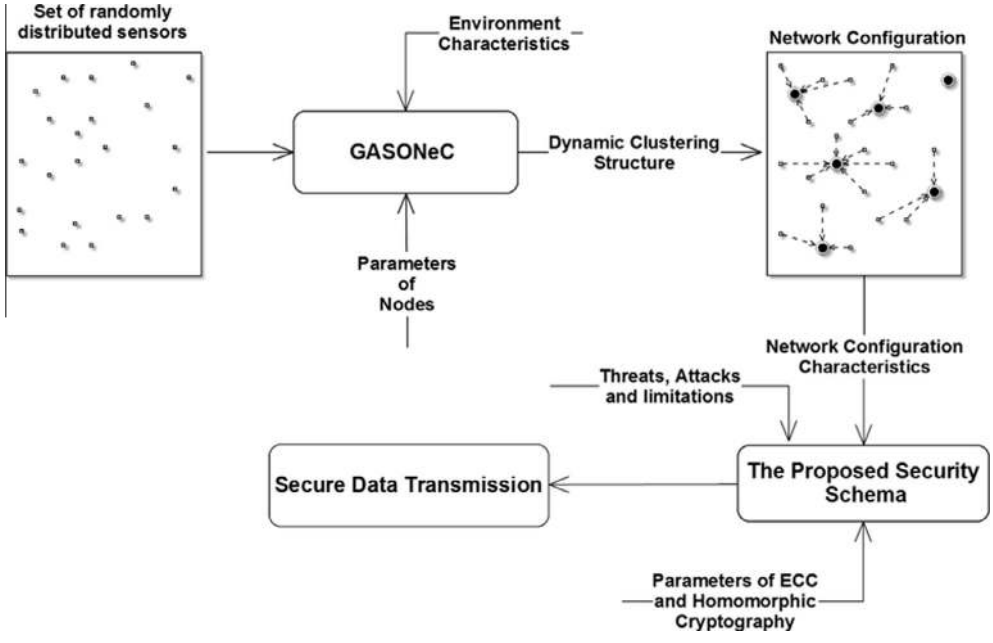

Fig. 1 General framework for creating our proposed secure data transmission algorithm.

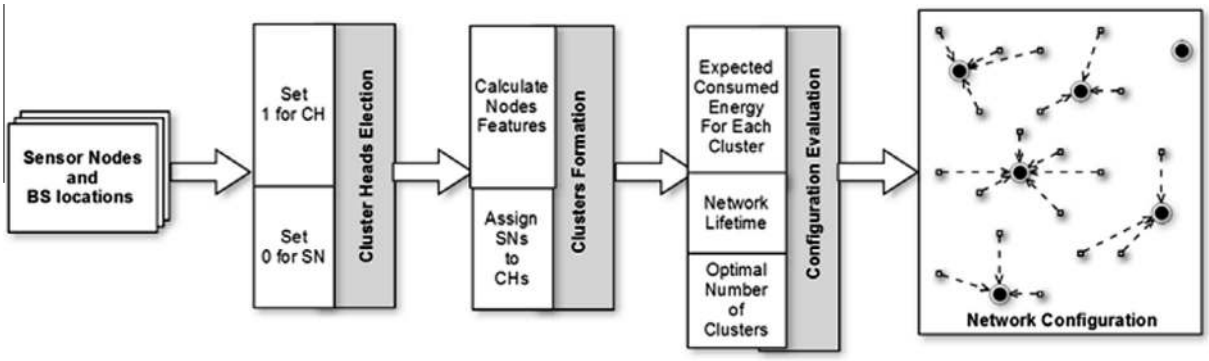

Fig. 2 GASONeR consists of three steps: cluster heads election, clusters formation, and configuration evaluation. the arrows depict the data flow.

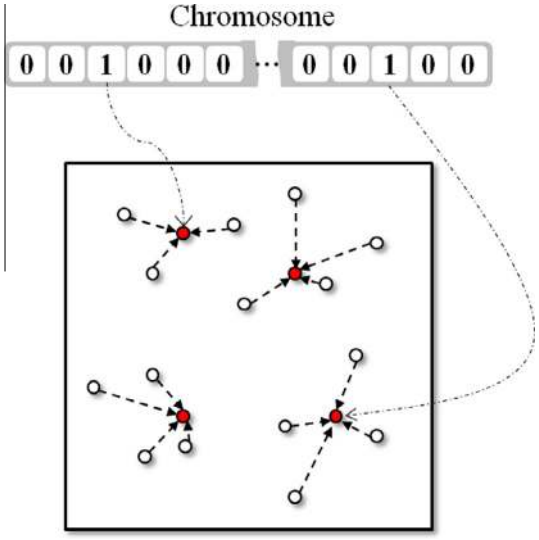

Fig. 3 GA chromosome and the mapping to network clusters.

The number of neighbors of a node i (donated by $N(i)$) is the count of nodes that far from i by a distance d less than or equal to θ. Where the θ value is estimated according to the network area.

$$N(i) = |n_j| \Longleftrightarrow d_{in_j} \leqslant \theta : (i \neq n_j \ \& \ \theta \ > 0) \tag{3}$$

So, the value of N_e is calculated as:

$$N_e = \frac{\sum_{i=1}^{|ch|} N(i)}{N} \tag{4}$$

In addition to fitness function, GASONeC calculates the probability of select $P(S_i)$, the expected count of select (EC), and the actual count of select for each chromosome as shown in Eq. (5):

$$p(s_i) = \frac{f(i)}{\sum_{x=1}^{n} f(x)} \tag{5}$$

where n is the number of the chromosomes in the population. A generation of the GA begins with reproduction. GASONeC selects the mating pool of the next generation using the weighted roulette wheel. The evolution terminates when one of the following criteria is satisfied: (1) the maximum number of generations is reached; or (2) the fitness converges. Upon completion of the GA evolution, the chromosome that gives the best fitness value is used to restructure the nodes.

4.2. Key establishment and plaintext encryption

A new encryption key is created in each transmission round for each sensor node. To overcome memory constraint, we designed the encryption key of 176-bit that includes the following components. The first component consists of 148 bits produced by the ECC and the ECC parameters are embedded in the sensors. The second component of the key represents the node ID in 13 bits. Although a typical size of a WSN is in the range of 200 nodes in most practical applications (Huang et al., 2014), a large scale WSN could consist of thousands of nodes. Hence, we allocate 13 bits for the node ID that is able to represent more than eight thousand sensors. The third component consists of 15 bits to encode the distance between the sensor and its CH $D_{i,CH}$. The seed key at sensor node i is the set to $\{K_i, ID_i, D_{i,CH}\}$.

To minimize the network communication overload, the authentication process is built upon the first part of the key. The key establishment between a SN and the BS is a two-party authentication process. The cluster heads (CHs) send the hashed ECC key from the member nodes to the BS to initialize the transmission sessions. We assume that every node knows its ID. The hashed code of the ECC key is produced using SHA-2 hashing function (Zhou et al., 2014).

To be self-content, we briefly review ECC encryption. Generally, an elliptic curve is a cubic equation with the following form:

$$y^2 = x^3 + ax + b, \tag{6}$$

where a and b are integers that satisfy $4a^3 + 27b^2 \neq 0$. The first step before data transmission between SNs, an elliptic curve and a base point p that lies on the curve must be known for every SN in the network. We assume that the elliptic curve parameters as well as the base point p are preloaded in the memory of each SN. To generate a shared secret key between SN A to SN B, A chooses a random prime integer k_A and B chooses a random prime integer k_B. k_A and k_B are the private keys of A and B, respectively. After that, Eqs. (7) and (8) are followed to generate the public keys $\bar{k_A}$ and $\bar{k_B}$ for A and B, respectively.

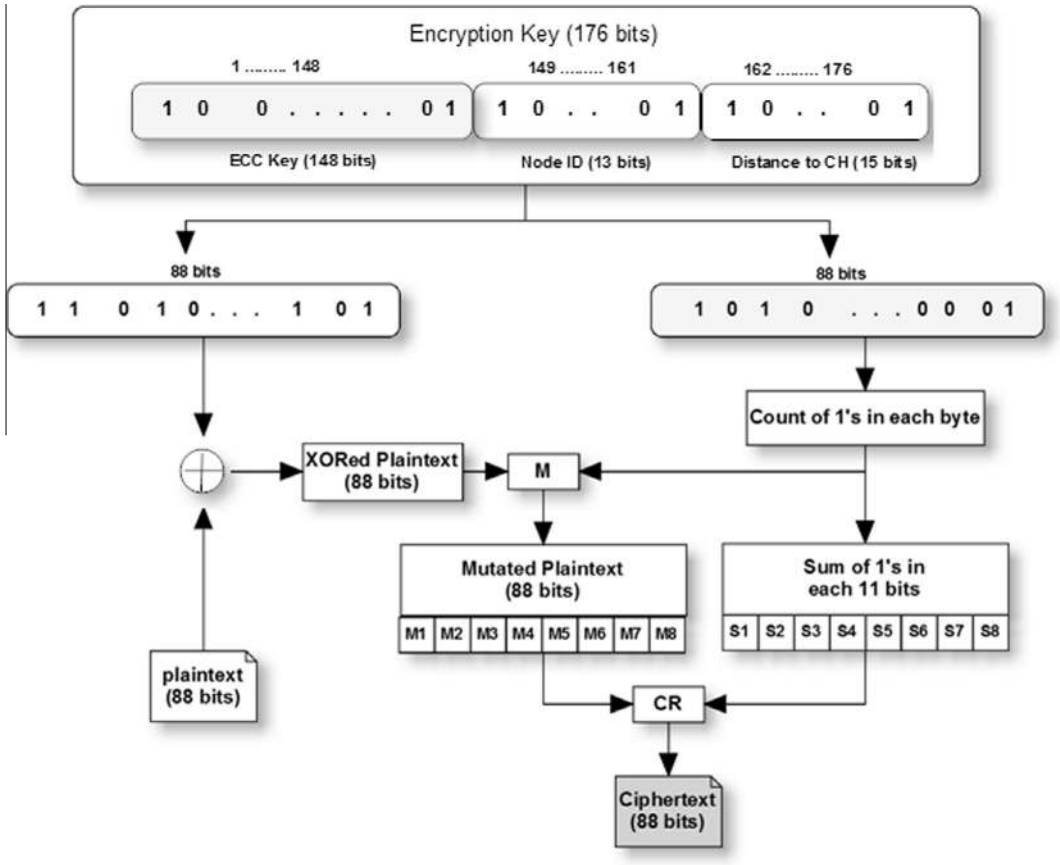

Fig. 4 The proposed encryption scheme.

P_1	01100100	01100001	01100011	00110101	00110011	01100011	00110001	00110111	01100011	00110010	00110101
Plaintext	01100011	01101100	01101111	01110101	01100100	00100000	01100011	01101100	01101111	01110101	01100100
XORED	00000111	00001101	00001100	01000000	01010111	01000011	01010010	01011011	00001100	01000111	01010001

P_2	00110000	01100110	01100100	00110100	01100100	00000000	00001000	00000001	11100000	00000000	01100100
Count(1's)	2	4	3	3	3	0	1	1	3	0	3
XORED	00000111	00001101	00001100	01000000	01010111	01000011	01010010	01011011	00001100	01000111	01010001
Permuted	01100111	00010101	00111100	01110000	01100111	01000011	10010010	10011011	00111100	01000111	01100001

Fig. 5 The permutation result for the plaintext 'cloud cloud' using an encryption key with two parts p_1 and p_2.

$$\bar{k_A} = k_A * p \tag{7}$$

$$\bar{k_B} = k_B * p \tag{8}$$

The public keys of both A and B are curve points. The private keys k_A and k_B specify the times of multiplying the base point p by itself to generate the public keys. After sharing public keys, they generate a shared secret key R, which needs to be the first part of our proposed encryption key, by multiplying public keys by their private keys as shown in Eq. (9). With the known values of $\bar{k_B}, \bar{k_A}$, and p, it is computationally intractable for an eavesdropper to calculate k_A and k_B which are the private keys of A and B. As a result, adversaries cannot figure out R which is the shared secret key (Houssain et al., 2012).

$$R = k_A * \bar{k_B} = K_B * \bar{k_A} \tag{9}$$

The SN uses the main operations of point addition and point doubling of the elliptic curve to generate its public key. The strength of an ECC crypto-system depends on the difficulty of finding the number of times p is added to itself to get this public key. This number represents the private key of the SN. The multiplication operation can be executed by adding a point along an elliptic curve to itself repeatedly. The addition operation for any two points $\omega(x_1, y_1)$ and $\vartheta(x_2, y_2)$ over an elliptic curve is given by Eqs. (10), and (11) with the assumption that $\omega + \vartheta$ is equal to a new point $\theta(x_3, y_3)$.

$$x_3 = \lambda^2 - x_1 - x_2 \tag{10}$$

Byte Index	1	2	3	4	5	6	7	8	9	10	11
(4,5)	01100111	00010101	00111100	01110000	01100111	01000011	10010010	10011011	00111100	01000111	01100001
(5,3)	1	2	4	3	5						
(3,3)	–	4	2	3	5	6					
(3,1)	–	–	3	2	5	6	7				
(1,3)	–	–	–	5	2	6	7	8			
(3,1)	–	–	–	–	2	6	8	7	9		
(1,3)	–	–	–	–	–	8	6	7	9	10	
(3,4)	–	–	–	–	–	–	–	7	10	9	11
Result	1	4	3	5	2	8	6	7	10	11	9
Concatenated	01100111	01110000	00111100	01100111	00010101	10011011	01000011	10010010	01000111	01100001	00111100
Ciphertext	gp<g⊥›C’Ga<										

Fig. 6 The final result of the encryption process after applying the Concatenation operation using the count of ones in each 11-bits of p_2.

$$y_3 = \lambda * (x_1 - x_3) - y_1 \tag{11}$$

where λ is calculated as the following:

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & \text{if } \omega \neq \vartheta \\ \frac{3x_1^2 + a}{2y_1} & \text{if } \omega = \vartheta \end{cases}$$

Fig. 4 shows the main steps of the encryption process which contains three different operations: XOR, permutation, and concatenation. The processes M and CR denote permutation and concatenation operations respectively. Permutation is a process of changing one or multiple bit values in a given bit string. Concatenation is a process of taking two parent bit strings and producing corresponding child bit string by interchanging selected parts of bit strings between the parents. The permutation process is applied to create random diversity in the ciphertext, whereas the concatenation operation is used to change the order of the mutated text or image data. The main benefit of using these two operations is that they introduce relatively fair diversity in the ciphertext.

Algorithm 1: Plaintext Encryption Process

1: Partition the plaintext into 88-bits N blocks $S = \{b_1, b_2, \ldots, b_N\}$.
2: Generate the 176-bits bit sequence β
3: Divide β to two equal parts p_1 and p_2, 88-bits each
4: Calculate the count of 1's for each byte $X[x_1, x_2, \ldots, x_8]$ as well as count of 1's for each 11 bits $Y[y_1, y_2, \ldots, y_8]$ in p_2
5: **for** $i = 1, 2, \ldots, N$ **do**
6: $\delta = p_1 \oplus b_i$
7: $\bar{\delta}$ =Perm $\{\delta, X[i]\}$, where Perm is the permutation function.
8: α =Con $\{\bar{\delta}, Y[i]\}$, where Con is the concatenation function.
9: $Cipher[i] = \alpha$
10: **end for**
11: Return the ciphertext $Cipher[c_1, c_2, c_3, \ldots, c_N]$, where c_i is the ciphertext of b_i

As shown in Algorithm 1, the working steps of the encryption process are as the following: First, we divide the bit sequence into 176-bits blocks and each block is subdivided into two 88-bit blocks. Then we calculate the number of 1s in each byte as well as the number of 1s for each 11 consecutive bits in the second half of the bit sequence. After that, we convert the plaintext into their corresponding binary codes and group them into blocks of 88-bits. This binary code is xored with the first block of the bit sequence since the additive cipher XOR is secure for the same length of key-stream. Then, the permutation process is performed on each byte of xored plaintext. For example, if the number of 1s in the first byte of the second half of random bit sequence is 7, we mutate the 7th and 8th number bits in the first byte of the xored plaintext. The permuted plaintext is further concatenated as shown to generate the ciphertext. This concatenation operation is done repeatedly so that each of the 11 bits performs the operation at least twice in order to add relatively fair diversity.

For example, Fig. 5 shows the result of permutation process using the plaintext 'cloud cloud' with assumption that the encryption key parts are p_1 and p_2. To get the result of concatenation operation, the count of ones in each 11-bits of p_2 is calculated as (4 , 5 , 3 , 3 , 1 , 3 , 1, 3). Fig. 6 shows the final result of the encryption process after applying the concatenation operation.

4.3. Secure data aggregation

In cluster-based WSN, a CH consumes more energy than a SN due to receiving, processing and retransmitting the data. This phase aims to avoid energy consumption of CH by allowing it to aggregate the encrypted data of their cluster members without having to decrypt them. As a result, the attacker won't be able to eavesdrop on the data from intermediate nodes, resulting in much stronger privacy than the traditional aggregation schemes. To do that, we use the addition property of

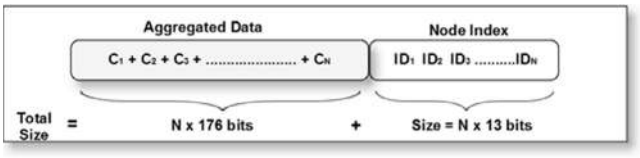

Fig. 7 The transferred ciphertext from CH to BS.

the homomorphic encryption that allows arithmetic operations to be performed on ciphertext.

For example, Let $E()$ denote encryption function. Let M be a group of messages under operation $\oplus$ and C is a group of messages under operation $\otimes$. $E()$ is a $(\oplus, \otimes)$ homomorphic encryption function if for $c_1 = E_{k_1}(m_1)$ and $c_2 = E_{k_2}(m_2)$, there exists a key k such that $c1 \otimes c_2 = E_k(m_1 \oplus m_2)$. In our scheme, each sensor node N_i senses data m_i, and encrypts it with k_i as shown in Eq.(12) and sends it to its CH. Where r is the round index in which the node produced the key k_i.

$$c_i = E(m_i, k_i^r, M) = m_i + k_i^r (mod M) \tag{12}$$

After receiving the sensed data, the CH collects n messages and aggregates them by simply adding them up as shown in Eq. (13):

$$c = \sum_{i=1}^{|N|} c_i = \sum_{i=1}^{|N|} m_i + \sum_{i=1}^{|N|} k_i^r (mod M) \tag{13}$$

The last step after aggregating the data is to forward it to the BS. In order to arrange the aggregated data, CH will attach all node indexes at the end of the message. Thus, the last form of the transferred ciphertext C_T to BS will be as shown in Fig. 7 with total size $N * 176 + N * 13$ bits, where N is the count of nodes in the cluster.

4.4. Plaintext retrieval process

Data decryption process aims to get back the original data that has sent by every SN in the network. Thus, the BS has to do two main processes to get the plaintext: data deaggregation, and plaintext retrieval. As shown in Algorithm 1. there are two main purposes of data deaggregation step. First, the BS has to extract the IDs of the nodes away from the aggregated ciphertext C. Since the cluster information is stored at the BS, it knows the count of nodes inside the cluster and their IDs. As explained in secure data aggregation phase, the BS will start the separation process at index $N * 176$ which represents the end of the aggregated ciphertext. Second, the reverse process of homomorphic encryption addition (Zhou et al., 2014) will be used to get $C_1, C_2, \ldots$ and C_N.

As soon as the BS separates the ciphertext, the plaintext retrieval process will start. The BS will use the shared keys forward by CH to decrypt the data and get the plaintext. The decryption process will start by generating the cryptography key for each SN. The working steps of the decryption process that aims to extract the plaintext are as the following: Similarly to the encryption process, BS divides the encryption key into 176-bit blocks and each block is subdivided into two 88-bit blocks. Then it calculates the number of 1s in each byte x as well as the sum of 1s for each 11 consecutive bits y in the second half of the random bit sequence. After that, it converts the ciphertext into their corresponding binary codes and groups them into blocks of 88-bits. Then, the ciphertext will be concatenated with y to generate the permuted ciphertext. After that, the permutation process is performed on each byte of the ciphertext using x. For example, if the number of 1s in the first byte of the second half of bit sequence is 7, BS permutes the 7th and 8th number bits in the first byte of the ciphertext. Finally, the BS will get the plaintext by X-Nored the first 88-bit block of the bit sequence with the result of the permutation process.

Algorithm 2: Ciphertext Decryption Process

1: By receiving ciphertext C with size b bits from CH x, get the count of its cluster members N
2: η = the first $(N * 176)$ bits of C, where η is the aggregated data.
3: $\mu = C$ - η, where μ is the cluster members' IDs starting from bit index $(N * 176) + 1$ of C
4: $\bar{\eta}$ = Deaggr (η), where Deaggr is the deaggregation function that revers the homomorphic operation
5: C_i =Assign $\{\bar{\eta}, \mu\}$, where C_i is the ciphertext of SN i
6: **for** $i = 1, 2, \ldots, N$ **do**
7: Partition C_i into 88-bits Q segments $S = \{s_1, s_2, \ldots, s_N\}$
8: Generate the 176-bits bit sequence β
9: Divide β to two equal parts p_1 and p_2, 88-bits each
10: Calculate the count of 1's for each byte $X[x_1, x_2, \ldots, x_8]$ as well as count of 1's for each 11 bits $Y[y_1, y_2, \ldots, y_8]$ in p_2
11: **for** $j = 1, 2, \ldots, Q$ **do**
12: α = Con $\{\bar{\delta}, Y[i]\}$, where Con is the concatenation function.
13: $\delta = p_1 \otimes s_j$
14: $\bar{\delta}$ = Perm $\{\delta, X[i]\}$, where Perm is the permutation function.
15: $Plain[j] = \bar{\delta}$
16: **end for**
17: Return the the plaintext $Plain[m_1, m_2, m_3, \ldots, m_Q]$, where m_j is the plaintext of SN_j
18: **end for**

5. Experimental results and discussion

The performance of the system was measured using the system throughput, network life time and the total energy consumption. In addition, we provide description about how our new schema prevents many kinds of attack to affect the network nodes.

5.1. Experimental settings

Our proposed method is simulated using the same characteristics of the sensor MICAz. The MICAz is based on the low-power 8-bit microcontroller ATmega128L with a clock frequency of 7.37 MHz and runs TinyOS as an event driven operating system. It also embeds a IEEE 802.15.4 compliant CC2420 transceiver with a claimed data rate of 250 kbps (demeulenaer et al., 2008).

In this evaluation, the WSN has the following properties:

- There is one base station that receives data from nodes;
- Nodes are stationary and their positions are known;

Table 1 Network properties.

Properties		Values
Number of Nodes		100
Initial node energy		0.5 J
Idle state energy		50 nJ/bit
Data aggregation energy		5 nJ/bit
Amplification energy	$d \geqslant d_0$	10 pJ/bit/m^2
(cluster head to base station)	$d < d_0$	0.0013 pJ/bit/m^2
Amplification energy	$d \geqslant d_1$	$E_{fs}/10 = E_{fs1}$
(node to cluster head)	$d < d_1$	$E_{mp}/10 = E_{mp1}$

- Provided with sufficient energy, each node can directly reach the base station;
- The characteristics and initial energy of each node are the same.

Table 1 lists the network parameters used in these experiments. In running GA to construct the network structure, we use the population size of 30 for 30 generations. The crossover probability and mutation probability are 0.8 and 0.006, respectively. The neighborhood distance δ is $20m$ throughout these experiments when LSD is calculated.

The average performance of 10 repetitions is reported. In each experiment, nodes are randomly placed in the field and the base station is also randomly placed at a certain distance to the field center. Comparison studies are conducted with different state-of-the-art methods to evaluate both security and efficiency of the network.

5.2. Image encryption

To evaluate the performance of the proposed method in case of images environment, we present the results of the histogram analysis and correlation analysis of two adjacent pixels in cipher images. First, the histograms of the plain image and its corresponding cipher image are shown at Fig. 8. From Fig. 8, it is noticeable that the histograms of the cipher and the plain images are significantly different. This big difference means that our proposed method avoids the statistical attack.

On the other hand, the correlation between adjacent pixels in an image is an important evaluation criteria for any encryption algorithm. This correlation may be in horizontal, vertical or diagonal directions. A secure encryption scheme should produce low correlation in the adjacent pixels in the cipher image. To evaluate our proposed secure schema, the correlation between two vertically adjacent pixels, two horizontally adjacent pixels, two diagonally adjacent pixels, and two anti-diagonally adjacent pixels in the plain and cipher images is calculated. This evaluation is done by selecting 1500 pairs of adjacent pixels from the plain image. Then, their correlation coefficient is calculated using Eqs. (14)–(16).

$$cov(x,y) = \frac{1}{n}\sum_{i=1}^{N}[E\{(x_i - E(x))(y_i - E(y))\}] \tag{14}$$

$$r_{xy} = \frac{cov(x,y)}{\sqrt{D(x) * D(y)}} \tag{15}$$

where x and y are gray-levels of two adjacent pixels in the image and r_xy is the correlation coefficients between two horizontally,vertically and diagonally adjacent pixels of these two images.

$$D(x) = \frac{1}{n}\sum_{i=1}^{N}[(x_i - E(x))]^2 \tag{16}$$

Table 2 Correlation analysis between the adjacent pixels (horizontal, vertical, diagonal, and anti-diagonal directions)

Input Image	Horizontally	Vertical	Diagonal	Anti-diagonal
Plain	0.9421	0.9513	0.8792	0.8968
Cipher	0.0018	0.0024	0.0052	0.0120

Table 3 Memory Requirements and CPU Cycles

Method	CPU (cycles)	Time (ms)	RAM (bytes)	ROM (bytes)
SkipJack	91224	12.353	292	7218
AES	68512	9.287	324	6994
LED	589652	78.972	378	5970
TWINE	128896	17.477	384	5280
BCC	91286	12.547	976	6240
Biswas	62396	8.547	542	5326
Our Method	66201	8.619	281	3845

Table 2 shows that there is a high correlation between the adjacent pixels (horizontal, vertical, diagonal, and anti-diagonal directions) in the plain image while the correlation between the two adjacent pixels in the cipher image is very low.

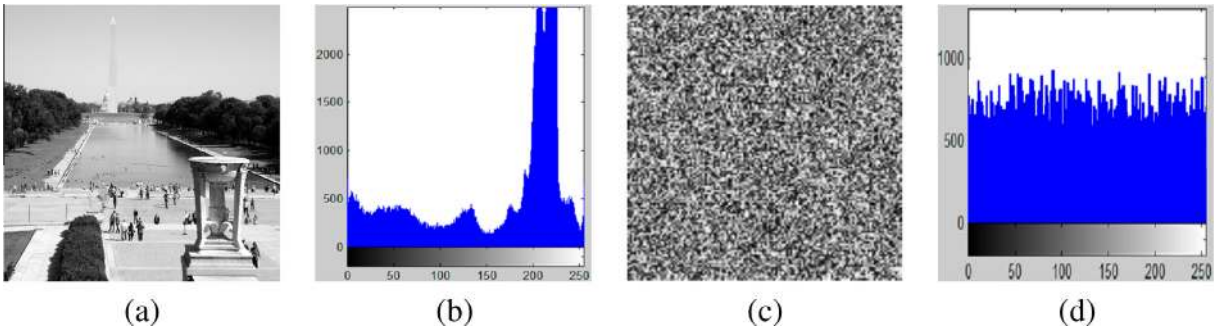

Fig. 8 Histograms of plain image and its corresponding cipher image. From left to right (a) the plain image, (b) histogram of the plain image, (c) the cipher image and (d) histogram of the cipher image.

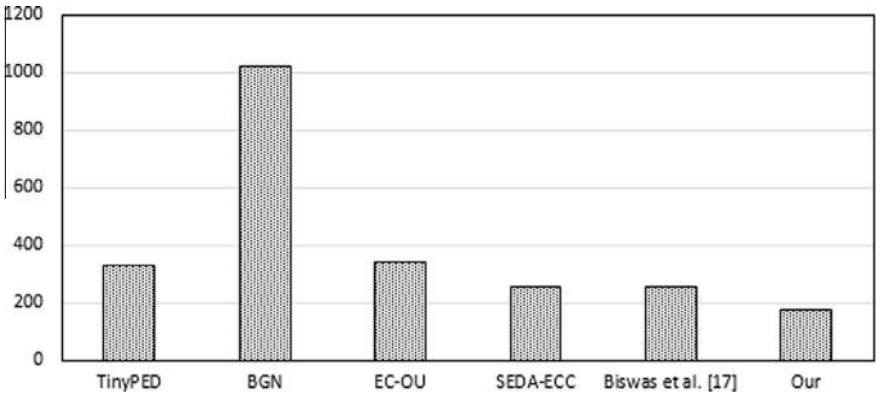

Fig. 9 Ciphertext size of different methods.

5.3. Memory requirements

Operation speed determines time complexity and is a significant factor for performance evaluation. Based on Biswas and Muthukkumarasamy (2014), ATEMU is used to get the total CPU cycles required to encrypt 32 byte data for MICA2 sensor mote. To evaluate the required memory size for each algorithm, TOSSIM has been used. Compared to state-of-the art methods, the results in the table indicate that our proposed algorithm performs better in terms of memory consumption (both RAM and ROM) as shown in Table 3. In case of CPU cycles, it can be seen that (Biswas and Muthukkumarasamy, 2014) is more efficient than the other protocols but still closed to our algorithm. Although our algorithm uses more time compared to Biswas and Muthukkumarasamy (2014), it is less than that of all other methods. Overall the proposed algorithm performed significantly better than other algorithms.

5.4. Communication overhead

The number of exchanged messages in each scheme is the same as each node needs to send two messages during the transmission process one HELLO message to establish the connection, and the other message for data transmission (Zhou et al., 2014). Thus, the communication overhead mainly depends on the ciphertext size of each algorithm if we assume that the number of message sending to the BS is the same. According to Zhou et al. (2014), the ciphertext size of SEDA-ECC is 256-bit. While EC-OU's ciphertext size is 341-bit, BGN's ciphertext size is 1,025-bit, and TinyPEDS's ciphertext size is 328-bit. In addition, the ciphertext size of the proposed algorithm in Biswas and Muthukkumarasamy (2014) is 256-bit. Our proposed method produces ciphertext with 176-bit size. Fig. 9 shows the comparison of ciphertext sizes. So, we can conclude that, the communication overhead of our proposed schema is better than other schemes.

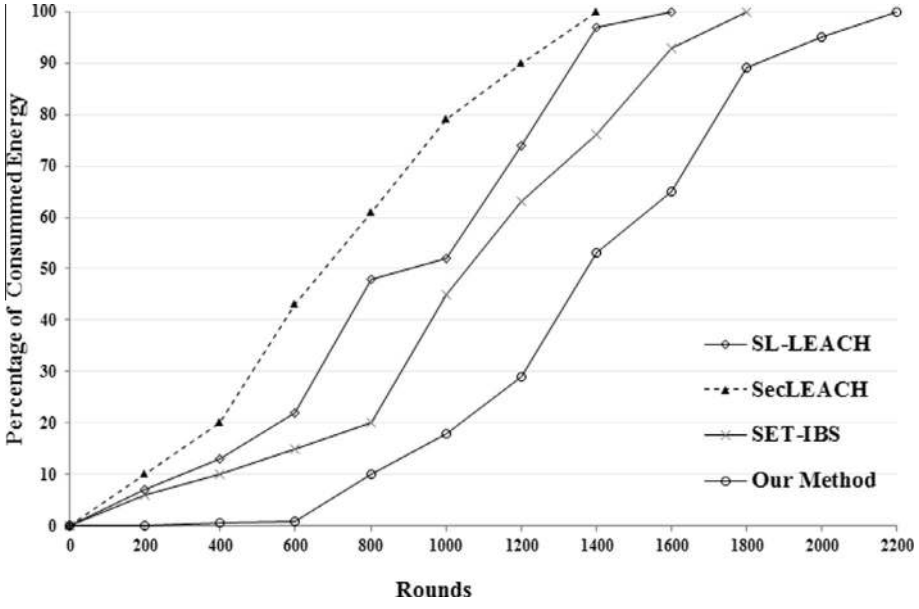

Fig. 10 Network energy consumption for different methods.

Table 4 Network energy consumption. Each column gives the percentage of the consumed energy for all sensor nodes at a specific round.

Round Index	0	200	400	600	800	1000	1200	1400	1600	1800	2000	2200
SL-LEACH	0	7	13	22	48	52	74	97	100	100	100	100
SecLEACH	0	10	20	43	61	79	90	100	100	100	100	100
SET-IBS	0	6	10	15	20	45	63	76	93	100	100	100
Our Method	0	0	0.5	1	10	18	29	53	65	89	95	100

5.5. *Energy consumption and network lifetime*

To evaluate the proposed schema from the energy consumption point of view, we compared it with SET-IBS, SL-Leach, and SecLeach methods as shown in Fig. 10. We find the proposed protocol has less power consumption and longest network lifetime. Compared with our methods, the increase of power consumption in the state-of-the art methods started before 200 rounds with the loss of the first node as shown in Table 4. As a result, the other nodes faced more load due to the increase of power consumption which reduced the network life. On the other hand, the proposed protocol lost the first

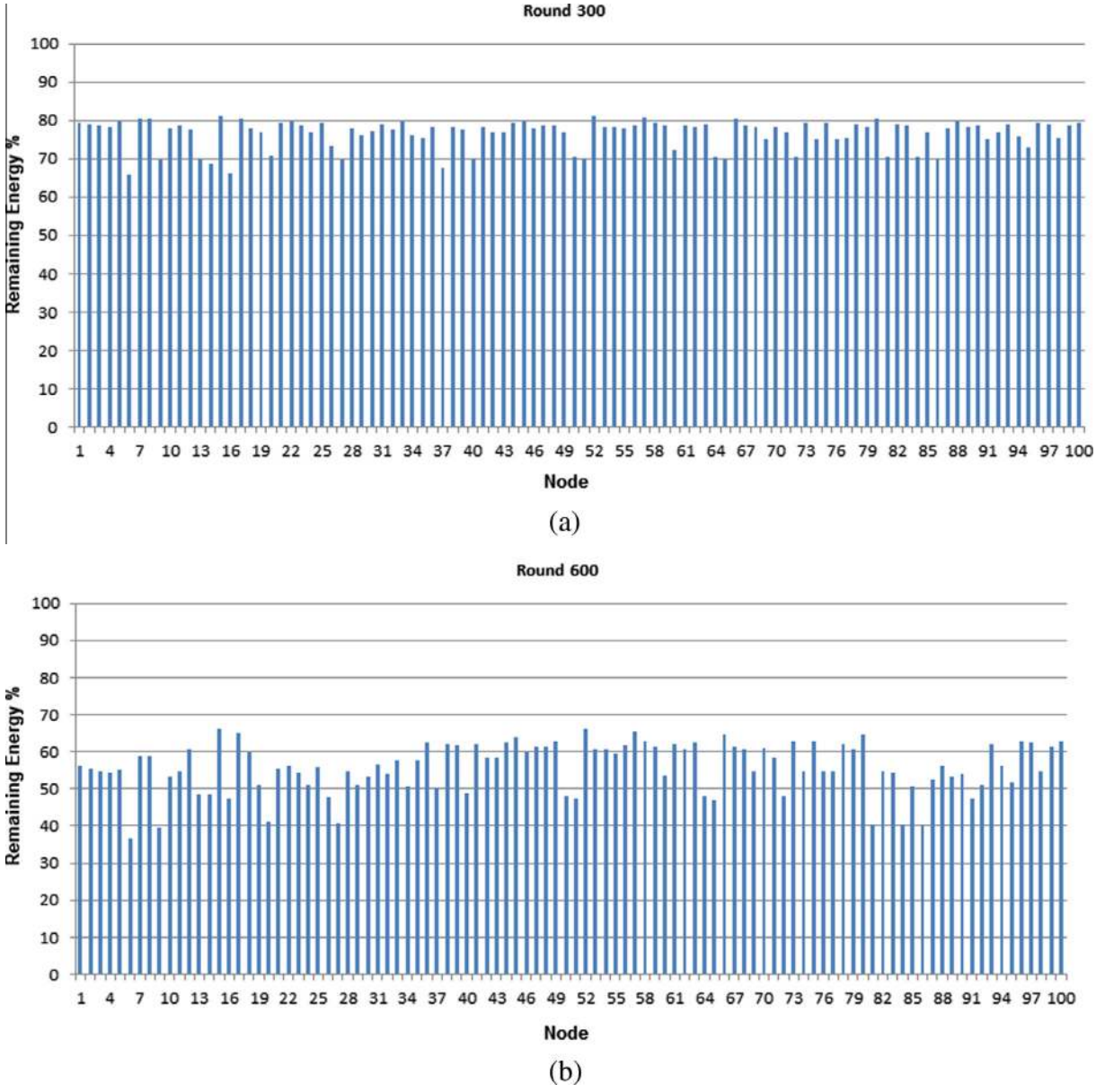

Fig. 11 The remaining energy of all sensor nodes (a) at network transmission round 100 and (b) at network transmission round 1000.

node at time. This reduced the power consumption and increased the network life time.

In order to evaluate the energy consumption, it is also interesting to examine the remaining energy of the sensor nodes. Ideally, we want to achieve the equal remaining energy in all nodes. Fig. 4 illustrates the percentage of the remaining energy of sensor nodes at transmission rounds of 100 and 1000. Nevertheless, it is clear that the remaining energy are fairly equal with some fluctuations. That is, as a consequence of our proposed method, the variance among remaining energy is quite low, which implies that the sensor nodes shared the burden of relaying messages and, hence, elongated the overall network life. (see Fig. 11)

Fig. 12shows the comparison between our proposed method and the state-of-the art methods in terms of network life time. In addition, Table 5 presents the percentage of live sensor nodes throughout the life span of the WSN using different methods. The number of round is the average of 10 experiments with random sensor node placement. As shown, our proposed method yielded the largest number of rounds when the first sensor node dies. It is clear that our proposed method greatly extended the network life.

5.6. Security analysis

We have tested our proposed encryption scheme against various security attacks. Here, we describe some of the important security analysis results including Passive attack analysis, key-space analysis, and compromised CH attack analysis.

5.6.1. Passive attacks

In the proposed protocol, the sensed data are encrypted by ECC and the homomorphic encryption algorithm which deals with eavesdropping. Thus, the passive adversaries cannot decrypt the eavesdropped message without the decryption key. Based on Huang et al. (2014), properties of the proposed schema settle the countermeasures to passive attacks.

5.6.2. Key space analysis

Key space size is determined by the total number of different keys used in the encryption scheme. The key-space of a good encryption algorithm should be large enough to make brute-force attacks infeasible. In our proposed encryption scheme, the set of secret parameters is $\{K_i, ID_i, D_{i,CH}\}$, where K_i is calculated based on $\{x_i, y_i, A, B\}$ which are the parameters of the curve. K_i is represented in 148 bits and has the following range $K_i \in (1, 2^{148}), ID_i \in (1, 2^{13})$, and $D_{i,CH} \in (1, 2^{15})$. Therefore, the complete key-space of the proposed encryption scheme is $3.4e^{38}$. Hence, we conclude that brute force attack is not feasible for such a large key space.

5.6.3. Compromised CH attacks

Compromised CH attack is an attack that tries to aggregate the data from all cluster nodes by deluding them that it is working as a CH. The aim of this attack is to analyze data and conclude specific information after receiving it. In our proposed schema, the role of CH is to forward the encrypted data from its cluster nodes and the BS without decrypting it. In addition, our encryption process depends on many factors, i. e, node location, and the distance between the node and its CH; this combination will require any attacker to have all information about the network operations and network topology (to know the distance between node and CH), and the secret key that is produced by ECC.

5.6.4. Sinkhole attack

Sinkhole attack adds a node to the network to capture all data as if it was the BS. The Sinkhole node has a strong power radio transmitter that allows it to provide a high-quality route by transmitting with enough power to reach a wide area of the network. In our method, the BS can detect the sinkhole attack as soon as it occurs. The BS starts the detection process by flooding the network with a request message containing the IDs and public key of the affected nodes after each round. The affected nodes reply to the base station with a message

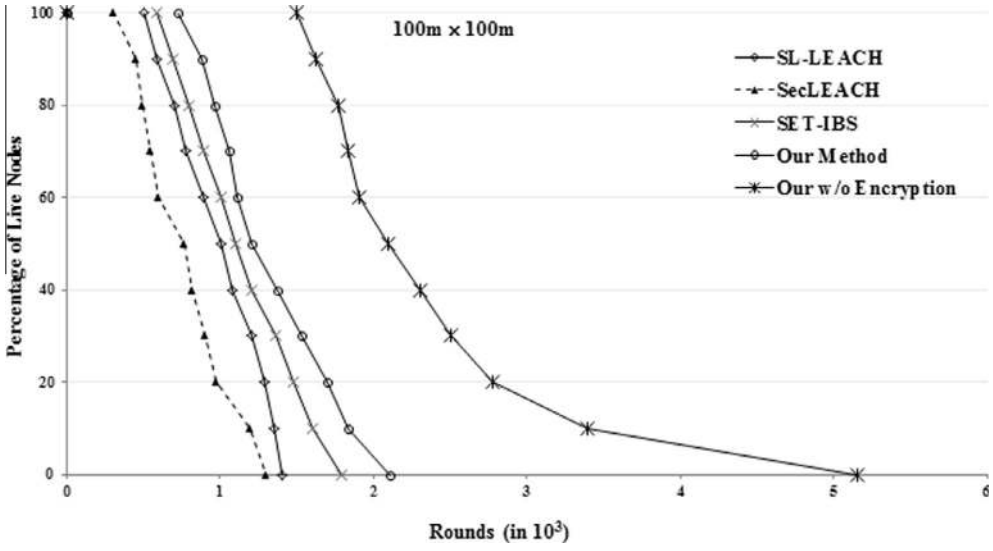

Fig. 12 Network lifetime figure that shows the percentage of the remaining number of live sensor nodes given the base station is placed at the corner of the field.

Table 5 Network life span. Each column gives the number of network transmission rounds when the respective percentage of sensor nodes becomes unavailable due to depletion of energy.

Percentage of alive nodes	100	90	80	70	60	50	40	30	20	10	0
SL-LEACH	509	592	698	780	893	1004	1082	1204	1290	1354	1402
SecLEACH	306	450	490	542	601	764	814	901	976	1200	1300
SET-IBS	591	695	801	894	1005	1100	1203	1360	1473	1600	1790
Our Method	723	885	965	1060	1110	1202	1370	1530	1707	1830	2103
Our w/o Encryption	1496	1620	1773	1832	1907	2100	2305	2501	2780	3400	5150

containing their IDs and the public keys. The received information is then used from the base station to construct a network flow graph for identifying the sinkhole. In other words, this attack is prevented by our method as the network structure is dynamically changed after each round. In addition the data are encrypted using a set of factors that are changed after each round and it is difficult for sinkhole attacker to get these parameters to retrieve the plaintext even if he can receive the data.

5.6.5. HELLO flood attack

HELLO flood attack sends the HELLO packet to the nodes, the node may assumes the attacked device as a neighbor that tries to connect with it. It aims to consume the network resources. In our method, the SN receives and transmits data to its CH only. In addition, it knows the distance between it and its CH after each round. So, we can conclude that the Hello Flood attack can be detected and avoided by the SNs. The same thing is applied in case of Selective Forwarding attack that tries to put a malicious node to act as normal node and drop the messages as soon as they receive it.

5.6.6. Denial of service attack

Also, the Denial of Service attack which sends unnecessary packets and utilizes more network bandwidth to prevent the user from accessing the service or resources is considered a popular attack. This attack is avoided also by changing the CH after each round and informing the CH with its cluster members. So, there is no direct connection between a SN and any other node except its CH which is well known by the SN. In addition, the acknowledge message that is sent from BS to the SN as soon as it receives the data will prevent this attack.

6. Conclusion

Forming network clusters is an effective way of improving scalability and longevity of WSN. However, security is a challenging issue in Cluster-based WSNs, since sensors are usually deployed with limited resources in unattended environments. Despite the great efforts in secure clustering of WSN, the dynamic nature of sensor network and numerous possible cluster configurations make searching for a secure and optimal network structure an open challenge. Due to its ability to provide high security with short key size, ECC is suitable for implementations on the devices with limited resources as sensor nodes in WSN. In this paper, we propose a novel encryption schema based on ECC and homomorphic encryption to secure data transmission in WSN with dynamic clustering nature. With the goal of optimizing the lifespan of the entire network, genetic algorithm is employed to search for the most suitable sensor nodes as the cluster heads to relay the messages to base station. Then, ECC is used to generate public and private keys for sensor nodes. The encryption key at each sensor node is 176-bits and is produced by combining the ECC key, identification number, and distance to its CH. To prevent the CH energy consumption as well as CH compromised attack, homomorphic encryption is used to allow CH to aggregate the encrypted data of its cluster members without having to decrypt them, to produce the final message that will be sent to the base station. Thus, it prevents the attacker from knowing anything even if the CH is compromised, because CH is not responsible to encrypt messages. Compared with other methods, our experimental results demonstrated that our proposed method greatly improves the network performance in terms of lifetime, communication overhead, memory requirements, and energy consumption. In addition, it prevents passive attack, CH compromised attack, and brute force attack.

References

Biham, E., Biryukov, A., Shamir, A., 2005. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. J. Cryptology 18 (4), 291–311.

Biswas, K.S.K., Muthukkumarasamy, V., 2014. An encryption scheme using chaotic map and genetic operations for wireless sensor networks. IEEE Sens. J. PP (99), 1.

Bogdanov, A., Khovratovich, D., Rechberger, C., 2011. Biclique cryptanalysis of the full aes. Adv. Cryptology 7073, 344–371.

Boneh, D., Goh, E., Nissim, K., 2005. Evaluating 2-DNF formulas on ciphertexts. In: the 2nd International Conference on Theory of Cryptography, pp. 325–341.

Cheng, Y., Agrawal, D., 2007. An improved key distribution mechanism for large scale hierarchical wireless sensor networks. Ad Hoc Networks 5 (1), 35–48.

Demeulenaer, G., Gosset, F., Standae, F., Vandendorpe, L., 2008. On the energy cost of communication and cryptography in wireless sensor networks, in: IEEE International Conference on Wireless and Mobile Computing and Networking and Communications, IEEE, pp. 580–585.

Elhoseny, M., Yuan, X., Yu, Z., Mao, C., El-Minir, H., Riad, A., 2014. Balancing energy consumption in heterogeneous wireless sensor networks using genetic algorithm. In: IEEE Commun. Lett. PP (99), 1.

Eryilmaz, E., Erturk, I., Atmaca, S., 2009. Implementation of skipjack cryptology algorithm for wsns using fpga. in: International Conference on Application of Information and Communication Technologies, IEEE, pp. 1–5.

Ganesh, S., Amutha, R., 2013. Efficient and secure routing protocol for wireless sensor networks through snr based dynamic clustering mechanisms. J. Commun. Networks 15 (4), 422–429.

Girao, J., Westhoff, D., Mykletun, E., Araki, T., 2007. TinyPEDS tiny persistent encrypted data storage in asynchronous wireless sensor networks. Ad Hoc Networks 5 (7), 1073–1089.

Houssain, H., Badra, M., Somani, T., 2012. Software implementations of elliptic curve cryptography in wireless sensor networks. J. Commun. Comput. 9, 712–720.

Huang, L., Jie, L., Guizani, M., 2014. Secure and efficient data transmission for cluster-based wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. 25 (3), 750–761.

Ibriq, J., Mahgoub, I., 2006. A secure hierarchical routing protocol for wireless sensor networks. In: IEEE International Conference on Communication Systems, IEEE, pp. 1–6.

Isobe, T., Shibutani, K., 2012. Security analysis of the lightweight block ciphers xtea and led and piccolo. Inf. Secur. Privacy 7372, 71–86.

Karakoca, F., Demircia, H., Harmancib, A., 2013. Biclique cryptanalysis of lblock and twine. Inf. Process. Lett. 113 (12), 423–429.

Kausar, F., Masood, A., Hussain, S., 2008. An authenticated key management scheme for hierarchical wireless sensor work. Adv. Commun. Syst. Electr. Eng. 4, 85–98.

Kumar, S., Jena, S., 2010. SCMRP: secure cluster based multipath routing protocol for wireless sensor networks. In: Sixth International Conference on Wireless Communication and Sensor Networks, IEEE, pp. 1–6.

Lalitha, T., Umarani, R., 2012. Energy efficient cluster based key management technique for wireless sensor network. Int. J. Adv. Eng. Technol. 3 (2), 186–190.

Li, B., Li, H., Wang, W., Yin, Q., Liu, H., 2013. Performance analysis and optimization for energy-efficient cooperative transmission in random wireless sensor network. IEEE Trans. Wireless Commun. 12 (9), 4647–4657.

Liu, Y., Tian, S., 2012. Design and statistical analysis of a new chaos block cipher for wsn. Commun. Nonlinear Sci. Numer. Simul. 17 (8), 3267–3278.

Mykletun, E., Girao, J., Westhoff, D., 2006. Public key based cryptoschemes for data concealment in wireless sensor networks. In: The IEEE International Conference on Communications, IEEE, Istanbul, pp. 2288–2295.

Oliveira, L., Ferreira, A., Vilaca, M., Wong, H., Bern, M., Dahab, R., Loureiro, A., 2007. Secleach-on the security of clustered sensor networks. Signal Process. 87 (12), 2882–2895.

Saminathan, A., Karthik, S., 2013. Development of an energy efficient and secure and reliable wireless sensor networks routing protocol based on data aggregation and user authentication. Am. J. Appl. Sci. 10 (8), 832–843.

Singh, M., Hussain, M., 2010. A top-down hierarchical multi-hop secure routing protocol for wireless sensor networks. Int. J. Ad hoc Sens. Ubiquitous Comput. 1 (2).

Suzaki, T., Minematsu, K., Morioka, S., Kobayashi, E., 2013. TWINE: a lightweight block cipher for multiple platforms. Sel. Areas Cryptography 7707, 339–354.

Xiao, W., Li, Y., Ke, C., 2005. SLEACH secure lowenergy adaptive clustering hierarchy protocol for wireless sensor networks. Wuhan Univ. J. Nat. Sci. 10 (1), 127–131.

XiaoJun, T., Zhu, W., Ke, Z., 2012. A novel block encryption scheme based on chaos and an S box for wsns. J. Chin. Phys. 21 (2).

Zhou, Q., Yang, G., He, L., 2014. A secure enhanced data aggregation based on ecc in wireless sensor network. Sens. J. 14 (4), 6701–6721.