



**A ROBUST KEY MANAGEMENT SCHEME FOR HIERARCHICAL  
WIRELESS SENSOR NETWORKS**

**ASO AHMED MAJEED AL-SALIHI**

**MAY 2015**

**A ROBUST KEY MANAGEMENT SCHEME FOR HIERARCHICAL  
WIRELESS SENSOR NETWORKS**

**A THESIS SUBMITTED TO  
THE GRADUATE SCHOOL OF NATURAL AND APPLIED  
SCIENCES OF  
ÇANKAYA UNIVERSITY**

**BY  
ASO AHMED MAJEED AL-SALIHI**

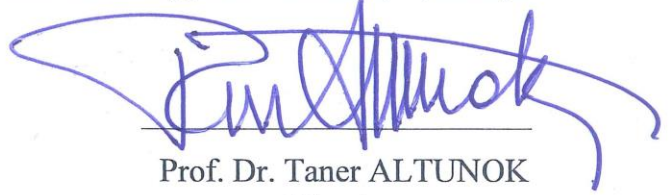
**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE  
DEGREE OF  
MASTER OF SCIENCE  
IN  
THE DEPARTMENT OF  
COMPUTER ENGINEERING**

**MAY 2015**

Title of the Thesis: **A Robust Key Management Scheme for Hierarchical Wireless Sensor Networks.**

Submitted by **Aso Ahmed Majeed AL-SALIHI**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.



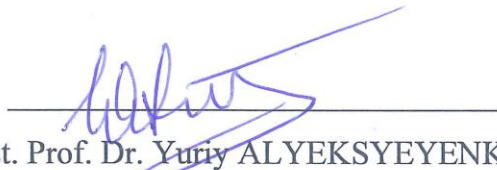
Prof. Dr. Taner ALTUNOK  
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

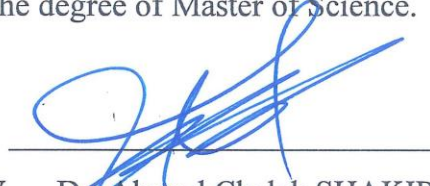


Prof. Dr. Müslim BOZYİĞİT  
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.



Assist. Prof. Dr. Yuriy ALYEKSYEYENKOV  
Supervisor



Dr. Ahmed Chalak SHAKIR  
Co-Supervisor

**Examination Date: 07.05.2015**

**Examining Committee Members:**

Assist. Prof. Dr. Yuriy ALYEKSYEYENKOV (Çankaya Univ.)


Assist. Prof. Dr. Abdül Kadir GÖRÜR (Çankaya Univ.)

Assoc. Prof. Dr. Fahd JARAD (THK Univ.)



## STATEMENT OF NON-PLAGIARISM PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Aso Ahmed Majeed, AL-SALIHI  
Signature :   
Date : 07.05.2015

## **ABSTRACT**

### **A ROBUST KEY MANAGEMENT SCHEME FOR HIERARCHICAL WIRELESS SENSOR NETWORKS**

AL-SALIHI, Aso Ahmed Majeed

M.Sc., Department of Computer Engineering

Supervisor: Assist. Prof. Dr. Yuriy ALYEKSYEYENKOV

Co-Supervisor: Dr. Ahmed Chalak SHAKIR

May 2015, 75 pages

Wireless Sensor Networks (WSNs) are used in many applications such as monitoring, target tracking and military applications, etc. A network is comprised of a large number of small, cheap, computational, limited energy devices which are called sensor nodes. These nodes are randomly deployed in open area to gather information. They communicate with each other via wireless media which are, however, vulnerable to being attacked by malicious nodes that may impersonate legitimate nodes. This makes the whole network liable to being hacked. Thus, security for WSNs is needed to keep any gathered data safe and to protect the communication links between sensor nodes. Key management plays a critical role in achieving security in a WSN. In this thesis, a robust key management scheme for hierarchical WSNs depends on, and exploits, the advantage of symmetric and asymmetric key cryptography. The proposed scheme uses the SHA-1 in HMAC not unlike a symmetric key for cluster formation and encryption of the private key via ECC, such as asymmetric key cryptography. Ultimately, the proposed scheme should be used to solve all these problems mentioned above.

**Keywords:** Wireless Sensor Network, Data Sequence, Security Information and Its Issues, ECC, Scalar Point Multiplication, Key Pre-Distribution.

## ÖZ

### HİYERARŞİK SENSÖR ŞEBEKESİ İLE İLGİLİ SAĞLAM BİR KİLİT YÖNETİMİ

AL-SALIHI, Aso Ahmed Majeed

Yüksek Lisans, Bilgisayar Mühendisliği Anabilim Dalı

Tez Yöneticisi: Yrd. Doç. Dr. Yuriy ALYEKSYEYENKOV

Eş Tez Yöneticisi: Dr. Ahmed Chalak SHAKIR

Mayıs 2015, 75 sayfa

Telsiz Sensör Şebekeleri (WSN) izleme, hedef takip etmek ve askeri alanlar gibi çok sayıda kullanım alanına sahiptir. Şebeke sensör düğümleri olarak adlandırılan çok sayıda küçük, ucuz, sayısal ve sınırlı enerji cihazları içermektedir. Bu düğümler bilgi toplama amacıyla açık alanlara tesadüfi olarak yerleştirilir. Bunlar; meşru düğümler gibi hareket edebilen zararlı düğümler tarafından yapılacak saldırılara karşı hassas telsiz medyası kanalıyla birbirleriyle iletişim halindedir. Bu; tüm şebekenin “hack”lenmesine yol açar. Böylece, bir araya getirilmiş bilgilerin güvenliğini sağlamak ve sensör nodları arasındaki haberleşmeyi korumak üzere WSNlerin güvenliğine ihtiyaç duyulmaktadır. Kilit yönetim WSN’ de güvenliğin sağlanması açısından kritik bir rol oynamaktadır. Bu tez hiyerarşik WSN’ deki sağlam kilit yönetim projesi bağımlıdır ve simetrik ve asimetrik kilit kriptolojisinin avantajlarından yararlanmaktadır. Önerilen proje küme oluşumu için simetrik anahtar olarak HMAC içerisinde SHA-1 kullanılmaktadır ve ECC benzeri asimetrik kilit kriptolojisi kanalıyla özel anahtar şifrelemektedir. Sonuç olarak, önerilen proje yukarıda bahsedilen tüm sorunları çözmek için kullanılmalıdır.

**Anahtar Kelimeler:** Sensör Şebekesi, Veri Sırası, Güvenlik Bilgileri ve Güvenlikle İlgili konular , ECC, Skalle Nokta Çarpma, Kilit Ön- Dağıtım .

## **ACKNOWLEDGEMENTS**

First and foremost, I would like to express my sincere gratitude to Assist. Prof. Dr. Yuriy ALEKSEYENKO for his supervision, special guidance, suggestions, as well as patience and immense knowledge through the development of this thesis. Furthermore, I would like to express my sincere thanks to Dr. Ahmed Chalak SHAKIR, the minor supervisor, who devoted his time and knowledge in the implementation of this project.

I express my deep appreciation to my wife for her kindness, patience and unlimited support throughout the study period as well as my special sincere gratitude to my family (mother, brothers, and sisters) for their valuable support.

Finally, I especially thank the Iraq Ministry of Higher Education and Kirkuk University for their help, which motivated me to continue with my study.

## TABLE OF CONTENTS

STATEMENT OF NON PLAGIARISM.....	iii
ABSTRACT.....	iv
ÖZ.....	v
ACKNOWLEDGEMENTS.....	vi
TABLE OF CONTENTS.....	vii
LIST OF FIGURES.....	viii
LIST OF TABLES.....	x
LIST OF ABBREVIATIONS.....	xi

### CHAPTERS:

<b>1. INTRODUCTION.....</b>	<b>1</b>
<b>1.1. Background on WSN.....</b>	<b>1</b>
<b>1.2. Literature View.....</b>	<b>6</b>
<b>1.3. Motivation.....</b>	<b>13</b>
<b>1.4. Problem Statement.....</b>	<b>14</b>
<b>1.5 Organization of the Thesis.....</b>	<b>14</b>
<b>2. SECURITY AND KEY MANAGEMENT IN WSNs.....</b>	<b>16</b>
<b>2.1. Security Issue in WSNs.....</b>	<b>16</b>
<b>2.2. Security Goals.....</b>	<b>16</b>
<b>2.2.1. Authenticity .....</b>	<b>16</b>
<b>2.2.2. Confidentiality .....</b>	<b>17</b>
<b>2.2.3. Integrity .....</b>	<b>17</b>
<b>2.2.4. Availability .....</b>	<b>17</b>
<b>2.2.5. Freshness .....</b>	<b>17</b>
<b>2.2.6. Resilience .....</b>	<b>17</b>
<b>2.2.7. Scalability .....</b>	<b>18</b>
<b>2.2.8. Connectivity .....</b>	<b>18</b>



2.3.	Attacks in WSN .....	18
2.3.1.	Sybil attack.....	18
2.3.2.	Sinkhole attack.....	19
2.3.3.	Wormhole attack.....	20
2.3.4.	Hello flood attack.....	20
2.3.5.	Clone attack.....	21
2.4.	Cryptographic Techniques and Key Management in WSN.....	21
2.4.1.	Symmetric key cryptography.....	22
2.4.2.	Asymmetric key cryptography.....	23
2.5.	Hash Function.....	23
2.6.	Message Authentication Code (MAC).....	24
2.7.	Elliptic Curve Cryptography.....	25
2.7.1.	Groups.....	27
2.7.1.1.	Finite group.....	27
2.7.1.2.	Cyclic groups.....	28
2.7.2.	Adding point in ECC.....	28
2.7.3.	Doubling point in ECC.....	29
2.7.4.	Scalar point multiplication in ECC.....	31
2.7.5.	ECDLP.....	31
2.8.	Key Management Goals.....	32
3.	THE ENHANCED DATA SEQUENCE METHOD FOR ECC CRYPTOSYSTEM EDSM-ECC.....	33
3.1.	Introduction.....	33
3.2.	Background on Data Sequence.....	34
3.3.	The Proposed Method.....	35
3.4.	Implementation of the Proposed Method.....	38
3.5.	Analysis of the Proposed Method.....	43
3.6.	Consequence .....	45
4.	SYSTEM IMPLEMENTATION AND RESULTS.....	46

4.1.	Introduction.....	46
4.2.	The Network Model.....	46
4.3.	Assumptions.....	47
4.4.	The Network Phases.....	47
4.4.1.	Pre-distribution phase.....	47
4.4.2.	Distribution phase.....	48
4.4.3.	Update phase.....	69
4.5.	Security Analysis and Results.....	70
4.5.1.	Security goals analysis.....	70
4.5.2.	Attacks analysis.....	72
5.	CONCLUSION.....	75
	REFERENCES.....	R1
	APPENDICES.....	A1
	A. CURRICULUM VITAE.....	A1

## LIST OF FIGURES

### FIGURES

<b>Figure 1</b>	Sensors size.....	1
<b>Figure 2</b>	Wireless sensor networks with sink and BS.....	2
<b>Figure 3</b>	Samples of wireless sensor applications.....	4
<b>Figure 4</b>	Symmetric key cryptography .....	5
<b>Figure 5</b>	Asymmetric key cryptography.....	5
<b>Figure 6</b>	Hash function cryptography .....	6
<b>Figure 7</b>	Sybil attack.....	19
<b>Figure 8</b>	Sinkhole attack.....	19
<b>Figure 9</b>	Wormhole attack.....	20
<b>Figure 10</b>	Hello flood attack.....	20
<b>Figure 11</b>	Clone attack.....	21
<b>Figure 12</b>	Security role.....	22
<b>Figure 13</b>	The Secure Hash Algorithm-1 (SHA-1).....	24
<b>Figure 14</b>	The MAC.....	25
<b>Figure 15</b>	Adding in ECC.....	28
<b>Figure 16</b>	Adding with $P$ equal $-P$ in ECC.....	29
<b>Figure 17</b>	Doubling point in ECC .....	30
<b>Figure 18</b>	Doubling point in ECC when $y_p=0$ .....	31
<b>Figure 19</b>	Fibonacci sequence.....	34
<b>Figure 20</b>	The points of the elliptic curve $E_{29}(-1,16)$ .....	38
<b>Figure 21</b>	Encryption by EDS-ECC algorithm.....	41
<b>Figure 22</b>	Decryption by EDS-ECC algorithm.....	43
<b>Figure 23</b>	A comparison in terms of bits no. to send hello.....	45
<b>Figure 24</b>	The hierarchical structure for the sensor network .....	47

## FIGURES

<b>Figure 25</b>	The <i>BS</i> setup.....	48
<b>Figure 26</b>	The <i>Ls</i> setup .....	48
<b>Figure 27</b>	The <i>CHs</i> setup .....	49
<b>Figure 28</b>	The sending of encrypted <i>Share_Key</i> from <i>BS</i> .....	49
<b>Figure 29</b>	The sending of encrypted <i>Share_Key</i> from <i>BS</i> algorithm.....	50
<b>Figure 30</b>	The sending of encrypted <i>Share_Key</i> from <i>BS</i> flowchart.....	50
<b>Figure 31</b>	The decryption of a <i>Share_Key</i> by each <i>CH</i> flowchart.....	51
<b>Figure 32</b>	The decryption of a <i>Share_Key</i> by each <i>CH</i> algorithm .....	52
<b>Figure 33</b>	The <i>CHs</i> that broadcast Hello Message .....	53
<b>Figure 34</b>	The <i>CHs</i> that broadcast Hello Message algorithm .....	53
<b>Figure 35</b>	The <i>CHs</i> that broadcast Hello Message flowchart .....	53
<b>Figure 36</b>	The <i>CH</i> selection by the <i>L</i> sensors flowchart .....	54
<b>Figure 37</b>	The <i>CH</i> selection by the <i>L</i> sensors algorithm .....	54
<b>Figure 38</b>	The <i>CH</i> selection by the <i>L</i> sensors .....	55
<b>Figure 39</b>	Each <i>L</i> sends its ID to selected <i>CH</i> algorithm .....	55
<b>Figure 40</b>	Each <i>L</i> sends its ID to selected <i>CH</i> .....	55
<b>Figure 41</b>	Each <i>L</i> sends its ID to selected <i>CH</i> flowchart .....	56
<b>Figure 42</b>	Decryption of <i>L</i> members messages by each <i>CH</i> flowchart ....	57
<b>Figure 43</b>	Decryption of <i>L</i> members messages by each <i>CH</i> algorithm....	58
<b>Figure 44</b>	The acknowledgment of <i>CH</i> to its member's algorithm.....	58
<b>Figure 45</b>	The acknowledgment of <i>CH</i> to its member's flowchart.....	59
<b>Figure 46</b>	The acknowledgment of <i>CH</i> to its members .....	59
<b>Figure 47</b>	The decryption acknowledgement by <i>L</i> sensors flowchart .....	60
<b>Figure 48</b>	The decryption acknowledgement by <i>L</i> sensors algorithm .....	60
<b>Figure 49</b>	Remote <i>L</i> sends Hello Message .....	61
<b>Figure 50</b>	Remote <i>L</i> sends Hello Message flowchart .....	61
<b>Figure 51</b>	Remote <i>L</i> sends Hello Message algorithm .....	61

## FIGURES

<b>Figure 52</b>	Decryption of hello message with Ack by neighbor <i>L</i> algorithm	62
<b>Figure 53</b>	The neighbor <i>L</i> sends its ID to remote <i>L</i> .....	62
<b>Figure 54</b>	Decryption of hello message with Ack by neighbor <i>L</i> flowchart	63
<b>Figure 55</b>	The remote <i>L</i> chooses strongest signal from neighbore <i>L</i> .....	64
<b>Figure 56</b>	The acknowledgment of <i>CH</i> to the remote <i>L</i> .....	65
<b>Figure 57</b>	Each <i>CH</i> sends a list of members' IDs to the <i>BS</i> .....	66
<b>Figure 58</b>	The acknowledgment of <i>BS</i> to the <i>CH</i> .....	66
<b>Figure 59</b>	The <i>BS</i> sends the encrypt <i>Private_Key</i> to the <i>CHs</i> .....	67
<b>Figure 60</b>	<i>BS</i> sends the encrypted <i>Private_Key</i> to the <i>CHs</i> algorithm .....	67
<b>Figure 61</b>	The <i>CH</i> 's Extraction of the <i>Private_Key</i> algorithm .....	68
<b>Figure 62</b>	The <i>CHs</i> send the encrypt <i>Private_Key</i> to its members .....	69
<b>Figure 63</b>	The connectivity comparison .....	71
<b>Figure 64</b>	The key ring (30 operations) for the seed key in term of second	73
<b>Figure 65</b>	The comparision proposed scheme with LEACH .....	74

## LIST OF TABLES

### TABLES

<b>Table 1</b>	NIST Recommended Key Sizes .....	26
<b>Table 2</b>	The Data Sequence Example .....	35
<b>Table 3</b>	The Generation of the EC Points for $E_{29}(-1,16)$ .....	39
<b>Table 4</b>	The Data Sequence After Circularly Right Shifting .....	40
<b>Table 5</b>	Encryptions in EDSM-ECC Method .....	44
<b>Table 6</b>	Encryptions in [25] Method .....	44
<b>Table 7</b>	A Comparison in Terms of no. of Bits and Digits .....	45
<b>Table 8</b>	Comparison in Terms of no. of Bits for Sending Hello.....	45
<b>Table 9</b>	Comparison in Terms of Security Goals .....	71
<b>Table 10</b>	Comparison in Terms of Attack Security .....	73

## LIST OF ABBREVIATIONS

WSN	Wireless Sensor Network
DARPA	Defense Advanced Research Projects Agency
DSN	Distributed Sensor Networks
WINS	Wireless Integrated Network Sensors
CH	Cluster Head
BS	Base Station
PCK	Public Key Cryptography
HSN	Hierarchical Sensor Network
SHA-1	Secure Hash Algorithm
RSA	Ron Rivest, Adi Shamir, and Leonard Adleman
ECC	Elliptic Curve Cryptography
ID	Identity
KDS	Key Distributions Servers
ECDH	Elliptic Curve Diffie-Hellman
ACK	Acknowledge
L	Low Sensor
PBC	Pairing Based Cryptography
GPRS	General Packet Radio Service
EC	Elliptic Curve
LEACH	Low Energy Adaptive Clustering Hierarchy
SCK	Self-Certified Key
KDC	Key Distribution Center
SKC	Symmetric Key Cryptography
PKC	Public Key Cryptography
SHA-1	Secure Hash Algorithm -1
MAC	Message Authentication Code

HMAC	Hash Message Authentication Code
NIST	National Institute of Standards and Technology
NSA	National Security Agency
ECDLP	Elliptic Curve Discrete Logarithm Problem
DSN	Distributed Sensor Networks
WSN	Wireless Sensor Network
M	Message
SK	Share Key
AKC	Asymmetric Key Cryptography
CA	Certificate Authority
HSN	Heterogeneous Sensor Network
DES	Data Encryption Standard
ECDLP	Elliptic Curve Discrete Logarithm Problem
DSA	Digital Signature Algorithm
$Q$	Public Key
$P$	Base Point
$k$	Private Key
$P_m$	Plain Message
$C_m$	Cipher Message
EDSM-ECC	Enhanced Data Sequence Method for ECC
DLP	Discrete Logarithm Problem
XOR	Exclusive OR

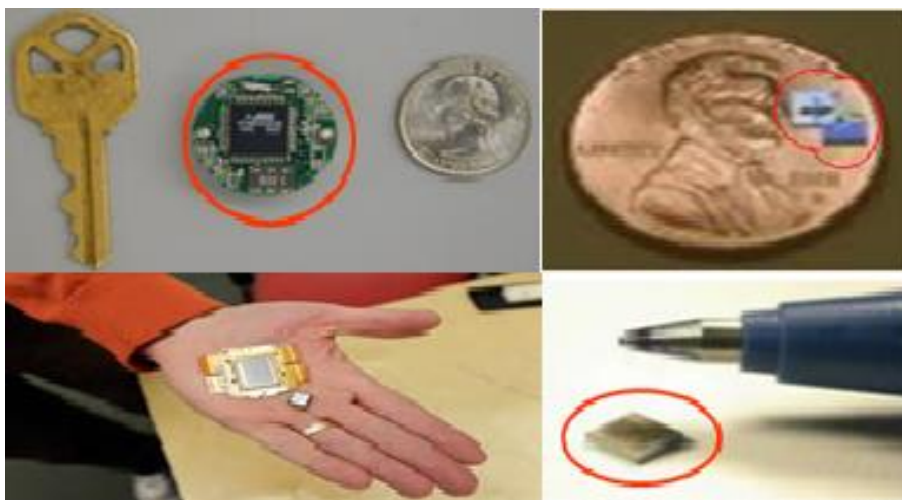


## CHAPTER 1

### INTRODUCTION

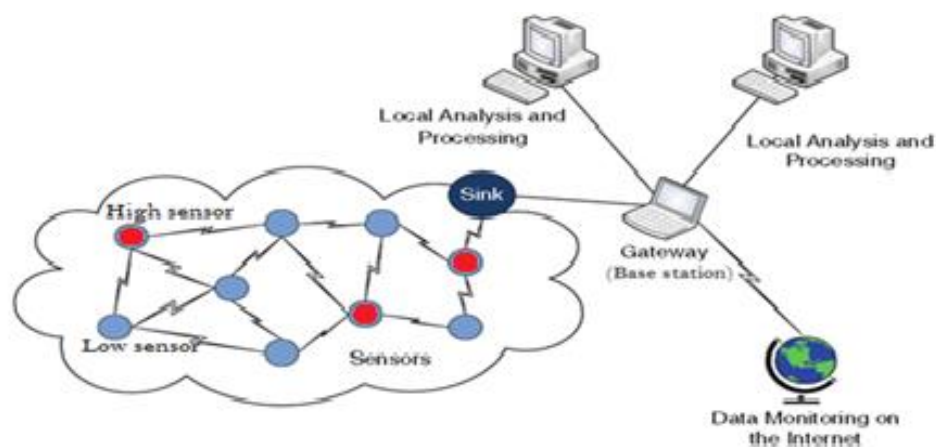
#### 1.1 Background on WSN

The wireless sensor has become a core component of modern life and its technologies have seen great development in the last decade. Moreover, the wireless sensor can be measured as one of the most major technologies of the 21st century [1]. In 1978, the Defense Advanced Research Projects Agency (DARPA) composed a workshop titled (DAR 1978). Moreover, in the early 1980s, it ran the Distributed Sensor Networks (DSN), which had remained a military area of focus on WSN research until the end of the 1990s, when the first motes were developed for environmental monitoring [2]. The concept of Wireless Integrated Network Sensors (WINS) was proposed by a collaboration of the Rockwell Science Center with the University of California in Los Angeles [3]. In addition, due to low sensor prices, very small sensor sizes and constraints on resources, they have been designed for a single purpose to serve one specific application [3].



**Figure 1** Sensors size

Fig. 1 shows sensors of different sizes and how small they can be. There are three types of WSN technology or systems. The first type is a non-propagation system. In this system, sensor nodes are connected to each other through wired media. For this reason, it does not support dynamic routing and it is manually configurable and highly deterministic in distribution cases. The second type is deterministic routing WSN systems. In this technology, there are both wired and wireless infrastructures. Furthermore, both play important roles in packet routing. The packet arrives at the wired infrastructure via several wireless hops. The final type of system is the self-configurable or self-enforcing system. They are non-deterministic in topological deployment and cannot manually configure the nodes. Furthermore, nodes communicate with each other via wireless infrastructure [4]. The self-enforcing characteristic of the sensors is deployed specifically in open areas or in uncontrolled areas. Our role is to organize the sensors in the absence of an infrastructure network. Moreover, they are expected to communicate with each other through wireless links. In this type of system, there are two types of infrastructure network architecture, namely a flat wireless sensor network and a hierarchical wireless sensor network [5]. The flat WSN consists of a base station, which is one or more distinctive elements of the WSN with far more computational, energy and communication resources. It represents a gateway between sensor nodes and the end user or system manager. Additionally, it has the ability to fuse data [5]. The sensors are the second component of a flat WSN. Furthermore, each sensor has identical properties with complexity management since each node represents a controller and a sensing part.



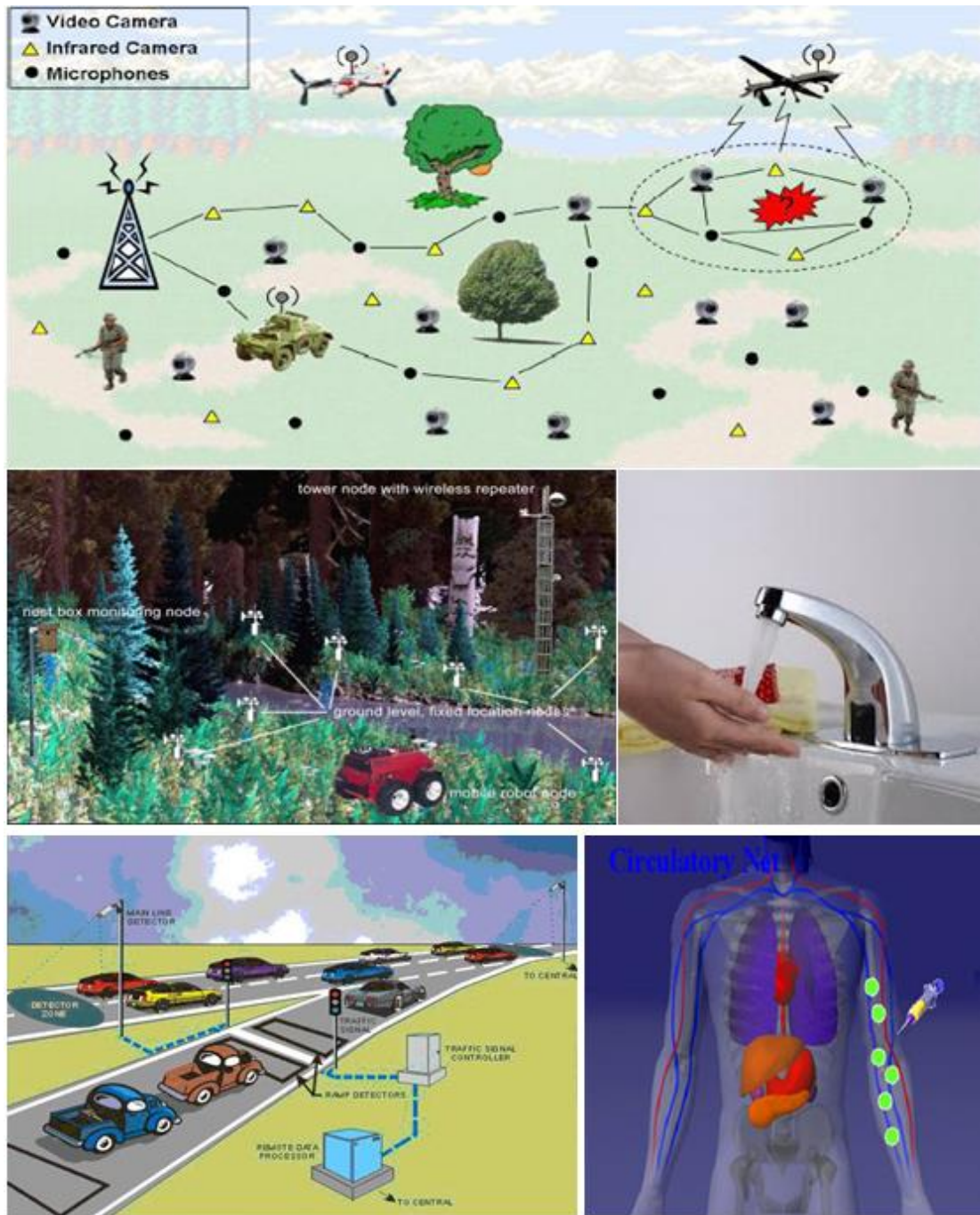
**Figure 2** Wireless sensor networks with sink and BS

However, in a hierarchical WSN, the network elements consist of a base station (*BS*), a cluster or a group, where the cluster leader acts as a high sensor called the cluster head (*CH*). The cluster head aggregates data from the other nodes in the same cluster and then sends it to the *BS*. The low sensor is another element of a hierarchical network which is a normal sensor with simple management and extends the lifetime of the network [5]. Moreover, there is a sink or pool with unlimited resources which exists in the network to reduce energy expenditure as well as to facilitate connections between nodes, as in Fig. 2 [2-6]. In recent years, we have seen rapid developments in wireless sensors which have become applicable in many fields, (such as Fig. 3) [2]:

- Environmental applications: home automation; traffic control; flood, tsunami or earthquake detection; microclimates and monitoring for hazardous gases; structural health monitoring.
- Military applications: targeting; terrain scanning; Intelligent Guiding; imaging and battlefield surveillance, which are a means to enable monitoring and tracking of movements of vehicles and the presence of enemy forces.
- Commercial applications: inventory control; vehicle tracking and detection; monitoring congestion and prevention of road accidents.
- Health applications: medical monitoring; behavior monitoring; elderly assistance; measurement of blood parameters; remote monitoring of physiological data; wearable computing.
- Agriculture applications: humidity and temperature measurement; automatic control over water sprinklers; precision agriculture.

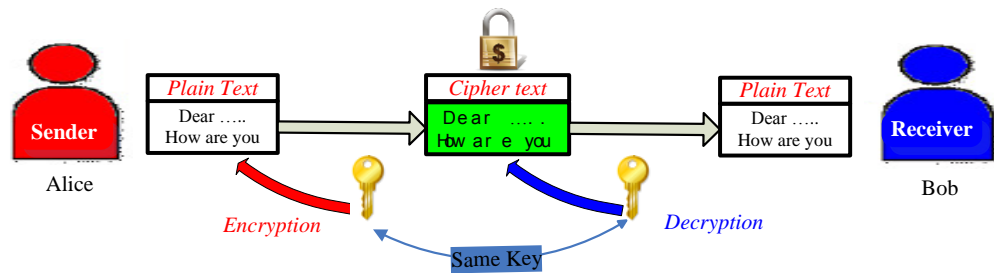
In previous studies and research in traditional networks that hide digital data, the authors used multimedia (image, audio and video), such as watermarking, which need complex algorithms, programs and unrestricted resources. However, in a WSN, due to restricted resources, these complex algorithms and programs cannot be used. Instead, it uses algorithms with less complexity than those of a traditional network. It is capable of being developed so that it protects the balance between security effectivity and restricted resources [6].

In WSNs, the sensor nodes connect to each other via IP addresses [7] or Key cryptography. Moreover, there are two kinds of key cryptography, one of which is symmetric and the other asymmetric or public key cryptography [8-9].



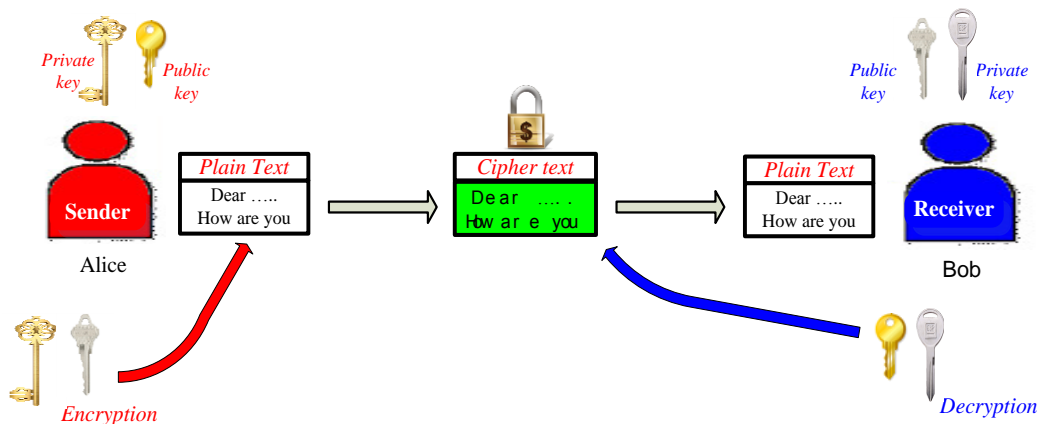
**Figure 3** Samples of wireless sensor applications

Symmetric key cryptography uses a single key, called a secret, which is used to encrypt and decrypt a message by the sender and receiver. In this sense, the sender and the receiver share the same secret key to exchange encrypted messages and to be able to decrypt them, as shown in Fig. 4 [8].



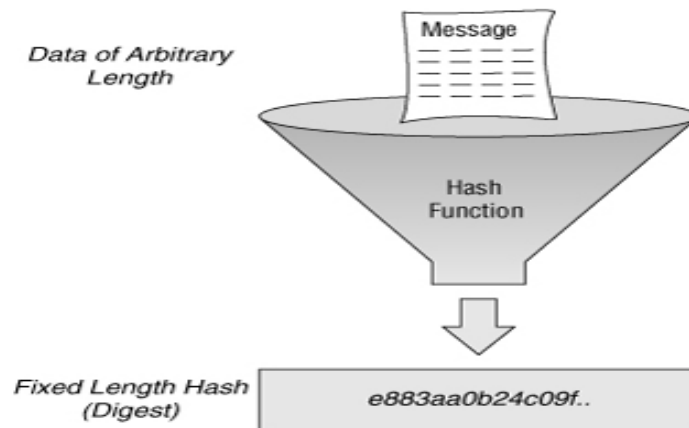
**Figure 4** Symmetric key cryptography

In public key cryptography (PKC), the sender/receiver have their own private keys, which are secrets, and a public key which is known to all nodes in the network. Furthermore, the sender/receiver use both the secret and the public key for encryption and decryption [9] as shown in Fig. 5. Recent studies have shown that using public key cryptography in Hierarchical Sensor Networks (HSNs) can enhance network performance, such as resilience, connectivity, etc. [9].



**Figure 5** Asymmetric key cryptography

Hash function (SHA-1) cryptography is an essential component of modern cryptography. It differs from symmetric and asymmetric cryptography, as shown in Fig. 6. Furthermore, it is defined as a deterministic procedure that generates a fixed-length bit string, called a *digest*. In Hash function cryptography, a message with any-length bit string is executed by taking inconstant bit patterns as an input, which then produces constant bit patterns of output. Subsequently, it is interesting to observe for an output message as it computes a unique digest [10-11].



**Figure 6** Hash function cryptography

An important element in the cryptosystem is the key and managing this key in the way of generation and distribution between the nodes in the network is not a trivial task. Furthermore, it has become a hot topic. Key management, therefore, is basic to confirm the security of communication of a wireless sensor network. Henceforth, how to establish an efficient key management in a WSN is a basic challenging problem due to the energy constraints, memory and computational processing capabilities of the sensor nodes [12].

## 1.2 Literature View

The security of a WSN has many challenges. These challenges include the wireless nature of communications, resource limitations on sensor nodes, routing topology prior to deployment, routing protocol, the type of communication, localization, data aggregation, operational environment and physical topology, the communication channel, size of the network and high risk of attacks to unattended sensors due to deploying it in an uncontrolled area without any knowledge about it. In recent years, several key management schemes have been proposed to meet the needs of different characteristics in a WSN which utilize cryptographic techniques for managing the key that is used to make the communication secure [6-9-13].

The cryptographic algorithms are necessary in sensor nodes to offer security services. Furthermore, public key cryptography consists of more than one algorithm, such as elliptic curve cryptography, Ron Rivest, Adi Shamir, and Leonard Adleman

(RSA) etc. The ECC has widely been used for security fields such as digital signatures and authentication. Moreover, the ECC is more appropriate for these small devices with constraints on resources [14].

Key management schemes can be ordered into three categories based on their encryption techniques: namely symmetric, asymmetric and hybrid [5]. We will review the literature related to each.

The key establishment technique should guarantee that the communication nodes in the network can verify the authenticity of the other nodes involved in a communication. The receiver node should recognize the assigned ID of the sender node [5].

In [15], Taogai Zhang and Hongshan Qu utilized the Hash function to reduce the effect of compromised sensor nodes on the uncompromised sensor nodes. The proposed scheme encompasses three phases. First, the key pre-distribution phase is carried out offline prior to deployment when the sensor nodes are in a defined area or target area through a Key Distributions Servers (KDS). A very large pool of  $w$  keys and their key identifiers called seed keys are generated the by KDS. Then the KDS selects an integer  $N$  as a security level. Additionally a Hash function  $H$ , is loaded into every sensor node via KDS and it randomly generates an integer  $m$ . which is between 0 and  $N$ . The  $m$  is then stored in the sensor node which is represented as a key security grade. The KDS randomly selects  $t$  seed keys from the key pool  $S$  and then performs on these ( $t$ )s the Hash function  $m$  times Hash. The second phase shares the key discovery. In this stage, each node discovers its neighbors via a common key identifier. The third phase is the path key establishment phase. In this phase, selected pairs of sensor nodes that do not share a common key are allocated a path-key which is not generated by the sensor nodes. Finally, the authors addressed that some compromised sensor nodes only affect part of uncompromised sensor nodes. Moreover, with the one-way Hash function, the attackers catch less key information from the compromised nodes.

In [16], Song Ju used a protocol that gathers ECDH with symmetric cryptography and a Hash chain. The protocol consists of 3 phases, one of which is the initialization phase, which occurs in the phase prior to the scatter of the sensor nodes in every node preloaded with same initial key. These include  $K_0$  to  $K_n$  where the  $K_n$  represents an initial trust as does the Hash chain function for every  $K$ . There is a time threshold



$T$ , and it is necessary to obtain key information from a compromise node with every node producing a public and private key via ECC. The next phase is the key establishment phase in which the two nodes implement the pairwise establishment key through the initial key. Moreover, there are two modes, one of which is new and the other old. The nodes which implement the key establishment are in a new mode and when finished, they turn into an old mode. After this phase, the node join phase saves the energy of the node by fully turning into the old mode and sleeping. When a new node joining begins, the new initial trust key will be  $K_{n-1}$  rather than  $K_n$ . In this protocol, the initial key and pairwise key are deleted constantly and this solves the signal attack. It furthermore reduces the computation and communication complexity via the initial trusted key. The author used ECDH with its public key, which solved the compromise attack.

In [17], Yiying Zhang and et al. distribute the keys on the basis of hop counts and a one-way function by the clustered architecture. The WSN model consists of *CH* sensors which gather the encrypted packets from their member nodes and then transport them to the *BS*. The member nodes transfer the encrypted packets from single or multi hops. Prior to deployment, each sensor is preloaded with an initial key which is used to cluster the formation after creation of the cluster. This would be deleted and a new key, which is an *ID* of the member, would be generated. Moreover, the sensor nodes select the *CH* not vice versa. Furthermore, the *CH* sends the acknowledge (*ACK*) to its member to tell its members that it is ready to receive its messages. Additionally, the authors divide the WSN area into levels, which are concentric circle and the *CH* sensors are stationed in the innermost circle. Each level is represented as a hop and the nodes share a common key for security within the same hop or level. However, in adjacent areas, they have duplicate keys.

In [18], Kishore Rajendiran et al. in their scheme, prior to deploying each sensor node, they preloaded it with a unique seed key which is an elliptic curve point as a private key and a statistically generated key ring through execution of ECC properties via point doubling and addition operations over the seed key. When pair nodes share identical private keys, the link is established between them. Additionally, the key ring increases the probability of sharing a common key. The scheme is resistant to Sybil and random attacks.



In [19], Yuan Zhang and et al proposed a scheme that causes cooperation between the *CH* sensors and sensor nodes and generates a group key within the cluster. Furthermore, only the *CH* sensors are responsible for generating and distributing the group key to the nodes within the cluster. During the initial key pre-distribution, the sink node selects the total number of groups. Additionally, during the cluster formation phase, each *CH* sensor preloads the unique *ID* of the other *CH* sensors. The *CH* sensors are selected and cluster formation is created with the *IDs* of each *CH*. Later, during the Hierarchical Key Generation phase, the hierarchical key is used for communication between the *CH* sensor and its member nodes within the cluster. Each *CH* sensor applies a one-way Hash function to the *ID* of the cluster member nodes. Next, the *CH* sensors distribute the keys to the nodes via a unicast message. The authors use this scheme to reduce the cost of communication to the sensor nodes.

In [20], Qing Yang, Qiaoliang Li and Sujun Li proposed a scheme based on public key cryptography in order to create distinct symmetric keys between neighbor nodes. Their network consists of high sensors with a *CH* sensor equipped with tamper resistance and low sensors, *L* sensors, as a cluster member. In the cluster formation, each *CH* sensor sends its *ID* via a hello message to the *L* sensors. When cluster formation is complete, the key establishment phase starts, which includes full pairwise keys preloaded into all high sensors prior to deployment. Thus, in order to create a secure link, communication between every *CH* sensor and *L* sensor, each *CH* sensor uses public and private keys which are preloaded prior to deployment in Rabin's scheme. Furthermore, each *L* sensor communicates with neighbor *L* sensors using a pairwise key. The authors' scheme includes sensor addition and revocation phases in order to save the network from adversaries in cases of deletion and to increase the network size in cases of adding. The authors' scheme decreases overheads of sensor node memory and computation in addition to achieving robust security against node compromise attack.

In [21], Sk. Md. Mizanur Rahman and Khalil El-Khatib proposed a novel key agreement based on Pairing Based Cryptography (PBC) over Elliptic curve (EC) to select secret points for heterogeneous WSNs. The authors assume a network which consists of high sensors which acting as a *CH* sensor equipped with tamper resistant hardware with properties exceeding those of *L* sensors. Each *CH* sensor and *L* sensor

is preloaded prior to deployment with the *ID*. The last element in the network is *BS*. The network runs a centralized distributed algorithm which means all *L* sensors send data to the *CH* sensors. In cluster formation phase each *CH* sensor broadcast hello message includes its *ID* and location. In other part the *L* sensors might receive one or more hello message. Then, after a/the period, each *L* sensor selects the strongest hello message as a cluster head *CH* sensor. If any *L* sensor has not received hello message, it connects to a *CH* sensor through other *L* sensors via the explore message. The routing in the network involves intra cluster and inter cluster. In the intra cluster phases, each *L* sensor is sent the information directly to the *CH* sensor or multi hop over other *L* sensors nodes thereby creating a tree form rooted at the *CH* sensor. On the other hand, in inter cluster phase, each *CH* sensor sends the aggregate information to the *BS* through the backbone of the *CH* sensor. The author's scheme is resistant against different attacks, including masquerade, wormhole, replay attack and message manipulation attacks.

In [8], Zhang Ying and Ji Pengfei proposed a scheme in a heterogeneous WSN. The proposed network consists of a *BS* with high-sensors becoming the *CH* and *L* sensors. The network elements are preloaded with a Hash function and a unique *ID*. Furthermore, the *BS* and *CH* sensors use light asymmetric key encryption dependent upon on the ECC and Hash function between the adjustment nodes in the same cluster. In the pre-distribution phase prior to deploying the network's elements, the system generates private and public keys via ECC for the sensors' nodes and the *BS*. Due to the *CH* sensors being equipped with tamper resistance hardware, they can use the same private and public keys. Each *CH* sensor is preloaded with public keys for every *L* sensor and its private and public keys. Each *L* sensor is preloaded with its own private key and the public key of all *CH* sensors. Therefore, in the cluster establishment phase, each *CH* sensor broadcasts a message containing its *ID* and its location. At the other side, each *L* sensor selects the best signal. Afterwards, each *L* sensor broadcasts its *ID* with the location to the selected *CH* sensor using the GPRS protocol. The link between the *CH* sensor and the *L* sensors was created through *share key* which was generated by ECDH. The author's scheme uses tamper resistance hardware in each *CH* sensor that prevents any adversary from gaining the key material when the *CH* sensors are compromised. Furthermore, it produced better

resiliency than the E-G method against the node that was captured. As additionally, the energy consumption is lower than the E-G method.

In [5], Ravi Kishore Kodali proposed a hybrid key management technique in which a hierarchical WSN with homogenous sensor nodes was used. Prior to deployment, each node is preloaded with an EC parameter. The ID-based private key is calculated by ECC, the Hash function, unique *ID*, list of *IDs* of other nodes and the master key of the *BS*. Additionally, the *BS* generates a pool with a large key and a randomly installed set of keys in each of the nodes. Moreover, each *CH* sensor was selected previously. After deployment, the *CH* sensors connect with the *BS* securely via an ID-based share key using bilinear pairing over the EC. Each *CH* sensor is connected to other *CH* sensors via the ID-based public key and Hash function. The *CH* sensors are propagated with the identifiers of a key from the key ring. If the nodes find an identical identifier, then it is used as a *share key*. If not, they are used as an ID-based key agreement. The *CH* sensors are changed periodically to save battery power. Then author's scheme provides a balance between security strength and resource consumption in HWSN. Furthermore, the energy overheads through using symmetric and asymmetric keys are minimized.

In [12], Md. Iftexhar Salam, Pardeep Kumar and Hoon Jae Lee proposed a scheme which relied on asymmetric key cryptography to establish secure key agreements (symmetric key) between nodes after they deployed in a WSN. Before deployment, the system manager preloads the public key with its corresponding private key in each node, unique *ID* and master public key. In addition, the *ID* of every node is identical to the master public key that is preloaded in the *BS* with a corresponding master private key. Only the *BS* has knowledge about the master private key. After deployment in neighbor discovery phase, the nodes exchange their public key with each other via swapping the hello message. Afterwards, every node sends the encrypted message to its neighbor by the public key of receiver, and only the receiver can decrypt the message with the receiver's private key. The nodes send an encrypted message to the *BS* by the master public key. Only the *BS* is able to decrypt the message as only it is knows the master private key. The *BS* sends the encrypted message to the nodes via the nodes' public key after sending those *IDs* to the *BS* for authentication. The authors' scheme is resistant to replication attacks such that whenever the *BS* finds any node having a high signal transmission range that is

greater than normal, it is immediately deleted and deleted if it checks the *ID* through monitoring the connectivity of each node. Thus, the scheme provides node authentication and perfect resilience.

In [22], W. Heinzelman and et al proposed the LEACH protocol, where the nodes in the network are homogenous (each node has identical characteristics) and each node has the ability to gather data in addition to data fusion in order to reduce the volume of information that must be transferred to the *BS*. After deployment, the sensors elect themselves to be the Cluster Head *CH* at any given time with a certain probability, after which they then broadcast their status to other nodes within the network. Other nodes select the *CH* sensor according minimum communication energy. Eventually, LEACH becomes a cluster-based protocol that distributes the energy load evenly among the nodes within the network through a randomized rotation of cluster heads (*CH*) within the cluster.

In [23], Rolf Blom proposed a key pre-distribution scheme in which any two nodes of a group find a pairwise key from a relatively small number of secret data. It creates its scheme by using one secret symmetric matrix and one public matrix. The Trent selects a public identifier which is a vector for each node and then each node will have the share of those matrixes. The nodes can calculate a share key between them through the share of those matrixes. The network includes  $n$  nodes and each node can access  $n-1$  keys. The disadvantage of this scheme is when an enemy compromises a sufficient number of nodes, it can recreate the complete key pool with a number of nodes being unable to communicate with any other node.

In [24], Canming, and Bao proposed a lightweight dynamic user authentication scheme which is based on SCK and modified by using ECC to establish a pair-wise key between sensor nodes and the user. The KDC selects a random number as its private key which is preloaded into each of the nodes followed by calculating the public key in each node through multiplying it by a base point. Moreover, it generates the *IDs* of every node and signature and saves the public key of the KDC in each entity before deployment. Each node knows the *ID* of the others and each node computes its own private key secret information. First, the new point named  $R$  which comes from random number multiplied by the base point is calculated. Second, the private key is computed via the KDS private key and is multiplied by the result of the Hash of its *ID* and X-coordinate of the point  $R$  added to the same

random number. Then each node computes its public key by multiplying its private key and base point. Each node establishes a pair-wise key through its private key, the other node public key and the base point. Eventually, this share pair-wise key is used to encrypt and decrypt the data, which is called MAC, and then broadcasts its new point  $R$ , its  $ID$  and the MAC. However, the scheme broadcasts the  $ID$  without any confidentiality in addition to there being no integrity because when any bits of the message changes, there is no way to verify it.

In[25], F. Amounas and E. H. El Kinani proposed a method to encrypt and decrypt the transmitted data by applying a data sequence to the output of the ECC with an order circular shift in order to reduce the number of bits transmitted. The authors' method converts each point to 8 bits to yield better performance from the data sequence. Our proposed scheme, named a robust key management scheme for hierarchical WSNs, will use this method to encrypt and decrypt the private key and the base point, which is the ECC parameter after enhancement, and decrease the number of transmitted bits to 6 bits rather than 8 bits. Moreover, we use the Hash function (SHA-1) which has low energy consumption with the ID to create a cluster formation. Finally, each node calculates the public key.

### **1.3 Motivation**

WSNs have emerged as an important new technology as integrated computing which is active and deals with the physical world. Security in WSNs has become an open research problem; therefore, it has attracted a huge number of researchers because its ubiquitous nature, randomly deployed consequentially random topology will be formed and self-enforcing. Moreover, a wide range of applications are now used in many civilian and military applications by deploying large numbers of small devices called sensor nodes, which have the ability to gather data and broadcast these data from their locations to the end user or system manager via a wireless channel. Since the nature of broadcasting over a wireless medium is vulnerable to adversaries who attack communication, securing these channels by using methods of security is required. With security, integrity, confidentiality and authentication can be achieved. The core of such security methods is known as a “*key*.” Managing these keys, called key management, is inevitable in WSNs. There are many key management schemes,

but there is no specific scheme appropriate for all sympathetic applications. The conventional key management techniques are infeasible to make the network reliable and both symmetric and asymmetric algorithms have their advantages and disadvantages. Therefore, our proposed scheme benefits from the advantages of symmetric algorithms (using the Hash functions with *ID*) and asymmetric algorithms (using data sequence on ECC).

#### **1.4 Problem Statement**

In actuality, the communications of the sensor nodes via a wireless medium is easily prone to various attacks. Since the sensor nodes are resource constraints, there is no robust security method to be used and the levels of security vary.

The proposed scheme strikes a balance between security and efficiency. Cryptography is a method to achieve security and the strength of the cryptography is commensurate with the strength of the security. Moreover, it is divided to symmetric and asymmetric key cryptography and both have their respective advantages and disadvantages. Consequently, our scheme benefits from the advantages of both. After deploying the sensor nodes in any area, the network must be connected in a secure way in order to obtain connectivity as well to ensure the safety of operations, confidentiality of sensitive data, and in order to prevent feeding the network with false information. The privacy of each node in the network depends on a small amount of information preloaded before deployment. Additionally, the network size may be increased by adding new nodes to make it scalable. Moreover, undesired nodes are revoked and provide resilience against attacks. To solve these problems our scheme utilizes the Hash functions (SHA-1) with a symmetric key based on ID and an asymmetric key based on applying a data sequence on the ECC.

#### **1.5 Organization of the Thesis**

This thesis contains five chapters. These sections illuminate all the necessary information for understanding and reading the thesis. Information and knowledge from previous chapters will be needed to be able to understand succeeding chapters.

Chapter 1 is an introduction to the history of WSNs, and contains a literature review, objectives and the problem statement of the thesis.

Chapter 2 includes security issues, attacks in WSNs, ECC, as well as the cryptographic and key management techniques in WSNs which are used in this thesis.

Chapter 3 includes the enhanced data sequence method for ECC cryptosystem which is used to sending private keys to cluster heads and low sensors.

Chapter 4 includes the implementation of a robust key management scheme for hierarchical WSNs and their results.

Chapter 5 includes the conclusion.

## CHAPTER 2

### SECURITY AND KEY MANAGEMENT IN WSNs

#### 2.1 Security Issue in WSNs

In general, in WSN applications, the network members (nodes) are scattered in an uncontrolled area with variable and unpredictable topologies. Moreover, the nature of the wireless transmission medium makes channel communication vulnerable and easily hacked by an adversary. This makes the WSN prone to various attacks. Furthermore, to protect data against these attacks, security is the only solution. Meanwhile, WSNs have resource constraints which significantly impact on, and prevent the utilization of complex security [26-27].

#### 2.2 Security Goals

The prime goal of security is to provide a security service, to protect data, to defend against different threats and to prevent the misuse of resources by a foe. Finally, it ensures authenticity, confidentiality, integrity, availability and freshness of data in addition to scalability and connectivity [26].

##### 2.2.1 Authenticity

Authentication enables each node in the network to ensure the identity of the peer nodes which communicate with each other so as to eliminate any fake messages and to guarantee that messages come from a trusted node. Symmetric Key Cryptography (SKC) or Public Key Cryptography (PKC) and also the Message Authentication Code (MAC) are utilized to achieve data and node authentication [26].



### **2.2.2 Confidentiality**

Confidentiality is an important stage to protect and keep any data secret during transmission between sensors and also between BS and sensors within the network by concealing messages and then avoiding the enemy. Unauthorized sensors cannot understand the messages, therefore, they remain secret. This is achieved with cryptography [26-28].

### **2.2.3 Integrity**

Integrity is of utmost importance to protect the data of received messages from alteration and modification by malicious nodes or adversaries. Until the network has confidentiality, there is still a possibility that data integrity or the message containing the data have been compromised by alterations or change. Integrity stops a malicious node present in the network from injecting bogus data. For this reason, MAC or Hash MAC (HMAC) is used to verify integrity [28].

### **2.2.4 Availability**

Availability ensures the survivability of sensor network services to authorized parties even though there are attacks within the network. It also ensures the updating of the security mechanism and is not influenced and bordered on the network performance [26].

### **2.2.5 Freshness**

Freshness ensures that all data and messages exchanged are modern and prevent the resending of old data. To prevent old messages from being sent again by an attacker, a timestamp can be added to the packet to achieve data freshness [26].

### **2.2.6 Resilience**

The number of keys which an attacker gains by physically capturing some nodes is

used by the attacker to hack the whole network and fail the system [27].

### **2.2.7 Scalability**

Scalability means that network size is flexible and its size can increase after deployment, which also ensures the level of security, stability and node properties when increasing the size of the network [26-29].

### **2.2.8 Connectivity**

Connectivity is the proportion of the probability of the nodes that are in contact with each other to create a network after deployment in order to ensure better performance of the network. In addition, whenever the percentage of connectivity is higher, the quality of network performance is finest [8].

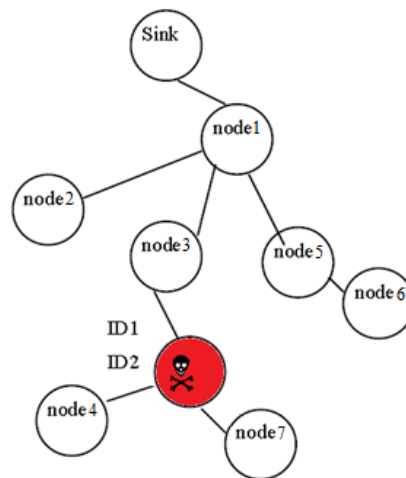
## **2.3 Attacks in WSN**

Due of any lack of security, attackers can intercept and read the content of any message. In addition, they can introduce false messages into the system via the network. Moreover, an attacker can capture a sensor and physically capture a node from which the attacker can steal the key material [30].

### **2.3.1 Sybil attack**

The attacking node is forged and it can disguise the identity of more than one node inside the network, thereby affecting data authenticity, confidentiality and data integrity. In addition, although identity fraud can negatively affect nodes and routing mechanisms used in the network, it appears as part of a direct legitimate path, as shown in Fig. 7. They can also send fake information into the system in the network. Moreover, it eavesdrops on passing messages and then attempts to learn the secret key and *ID* of the legal nodes. Furthermore, an attacker can initiate attacks

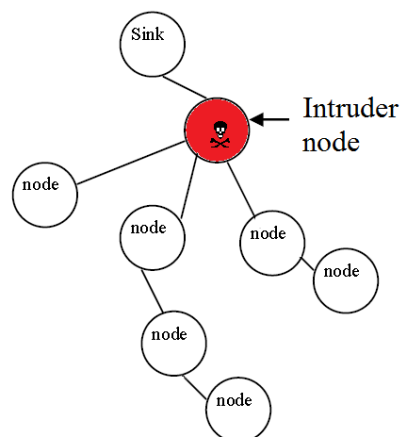
simultaneously or incrementally in several places. However, an authentication and cryptography mechanism can prevent such attacks [31-32].



**Figure 7** Sybil attack

### 2.3.2 Sinkhole attack

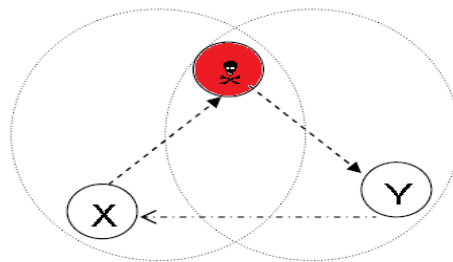
In general, malicious node disguises the victim nodes and is located near to the *BS* and persuades the victim nodes to send their messages to the *BS* at high power, as shown in Fig. 8. Additionally, it may be likened to a black hole absorbing everything passing into it. Moreover, it disrupts the link between the intruder node and the remaining nodes within the WSN; therefore, the WSN is deprived of its ability to provide service. Finally, the throughput of all the nodes around the malicious node decreases, and therefore, efficiency and performance of the network will decrease [30].



**Figure 8** Sinkhole attack

### 2.3.3 Wormhole attack

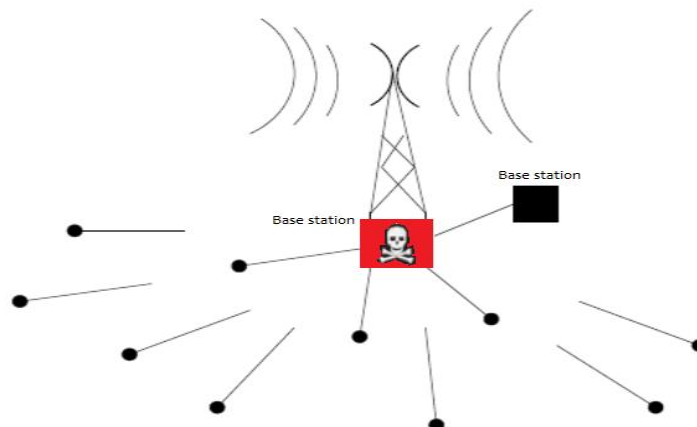
The malicious node uses a tunneling technique to establish itself between nodes in order to confuse the routing protocol. The malicious node disguises the victim nodes and shows itself as having higher communication resources than normal nodes in order to establish the best communication channels between them, as shown in Fig. 9. Moreover, wrong routing of information leads to changes in the network topology and the stream of messages will change. It can alter the packet or modify it, thereby damaging the packet [30].



**Figure 9** Wormhole attack

### 2.3.4 Hello flood attack

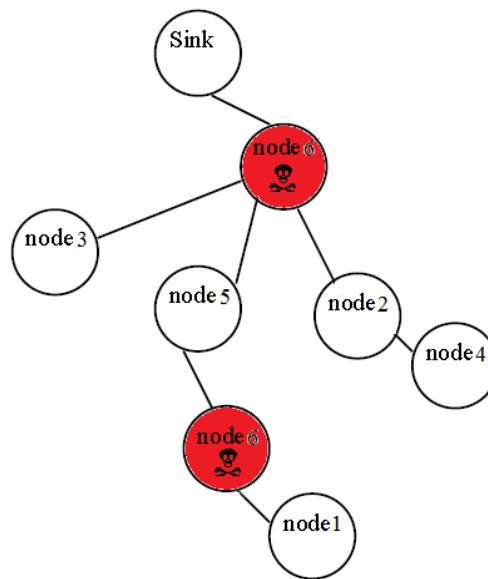
The intruder is a sink or BS, as shown in Fig. 10 and it broadcasts a hello message with strong transmission range and power to the network and acts as a fake sink to send their messages to it rather than to the legal sink. It disguises the BS by acting as a neighbor with many nodes in the network, thereby badly mixing the network routing [30].



**Figure 10** Hello flood attack

### 2.3.5 Clone attack

The attacker is a node. Its content comes from a copy of the content of another node captured physically. Additionally, the cloned node carries the ID of the victim node and is able to fabricate routing information within the network. It has the ability to access confidential information and secret keys. Moreover, in clone attacks, the malicious node could be more than one node with the same identity, as shown in Fig. 11, unlike a Sybil attack in which they appear as one node having different identities [33].

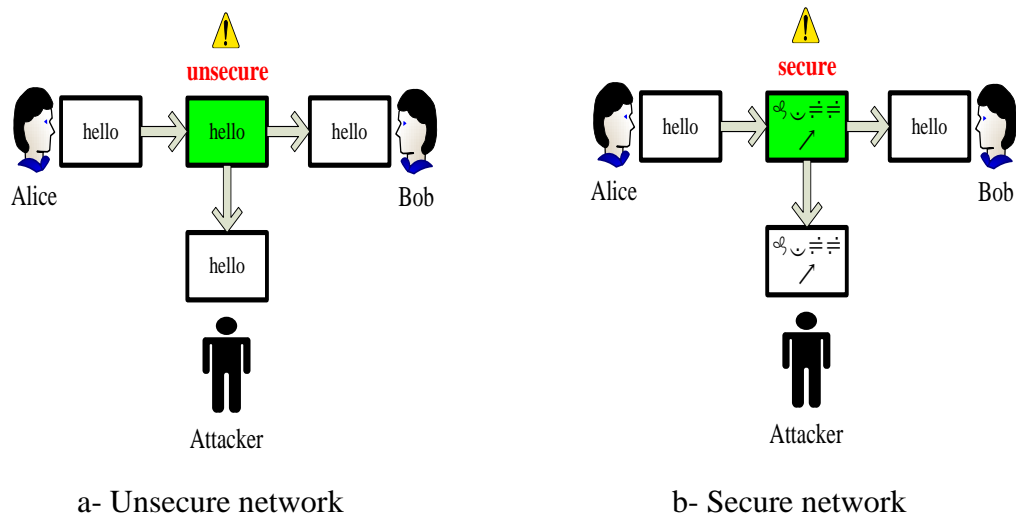


**Figure 11** Clone attack

### 2.4 Cryptographic Techniques and Key Management in WSN

No one can deny the importance of security in data communications and networking. Moreover, cryptography makes a network more secure and fortified against attacks and threats; therefore, network security is achieved through cryptographic mechanisms. Additionally, cryptography is the design and analysis of mathematical techniques that make communications secure by providing random sampling mechanisms and interactive proofs to allow the receiver's node to verify the delivered data by user nodes in the presence of malicious adversaries. Fig. 12 shows the differences between cases of a secure message and an unsecured one.

Cryptographic systems are divided into two categories: *symmetric* and *asymmetric* key cryptography, both of which are maintained via key management, which is a set of processes in which keys are created, stored, protected and transferred so as to create secure links between authorized nodes in the WSN in order to exchange reports in securely [34-35].



**Figure 12** Security role

### 2.4.1 Symmetric key cryptography

In symmetric key cryptography, both parties (sender/receiver) use the same key, which is secret. The sender uses this secret key to encrypt and convert the data, and then convert the message to a cipher text after which the cipher text is broadcast as a message. When the message is delivered to the receiver node, it uses the same secret key to decrypt the cipher message after it extracts the data. Here, the enemy cannot acquire the cipher text and he cannot decrypt it without the secret key, as in Figure 4. Examples of symmetric key cryptography include Data Encryption Standard (DES), Advanced Encryption Standard (AES), MAC, and HMAC. In addition, there are many advantages of encryption in symmetric key cryptography. It is perceived to have an extensive history, the size of key short, data throughput rate is high, and implementation of SKC is speedier than PKC. Moreover, ASC has the disadvantage such that both the sender and receiver must always keep the secret key and the key is easy to manage [34].

### **2.4.2 Asymmetric key cryptography**

The idea of asymmetric key cryptography, or Public Key Cryptography (PKC), as depicted in Figure 5, was announced in 1975 by Diffie, Hellman and Merkle. Asymmetric key cryptography comprises the utilization of two keys: the private key and the public key. A secret or private key, which is secret, needs to decrypt and sign transmitted messages. In addition, each node has only an isolated private key and is different from the other nodes. Moreover, only one node retains and knows this key and no other. In public key cryptography, each node knows this key and they need the public key to communicate with each other, either to encrypt a message transmitted to another node or to decrypt a received message to the node and verify the node's signatures. Furthermore, the public key is announced to the all nodes within the network. However, the two keys in public key cryptography are mathematically linked, one of which is used for encryption and the other for decryption. Eventually, the PKC examples are RSA, Diffie Hellman, ECC, El-Gamal [9-34-36].

### **2.5 Hash Function**

The Hash function is a basic part of cryptographic algorithms which does not use a key. In 1993, it was published by the National Institute of Standards and Technology (NIST) and was designed by the NSA (National Security Agency). In 1995, it was replaced by SHA-1, which was based on the principle of MD5 [36]. It is a one way function and when a message is input, it generates a unique message digest with a fixed length of 160 bits without the ability to inverse the output, which is the digest message to obtain the original text message, implying that it cannot be determined. Furthermore, the output of the Hash function SHA-1 is not equal to other digest messages, thereby implying that no two messages have the same output [37]. In addition, the size of the digest message is 160 bits for each message, as shown in Fig. 13 [36]. This is called the discrete log Hash function. Finally, it provides integrity data authentication [38].

- 
1. Begin with a message  $M$  suffixed bits, as defined in the text, to find a message  $P$  of the form  $P=M_1||M_2||M_3||\dots||M_n$  and every  $M_i$  consist of 512 bits.
  2. Set  $H_0 = 67452301$ ,  $H_1 = \text{EFCDAB89}$ ,  $H_2 = \text{98BADCFE}$ ,  $H_3 = 10325476$ ,  $H_4 = \text{C3D2E1F0}$ .
  3. for  $i = 0$  to  $n - 1$ , run following:
    - 3.1 Write  $M_i=W_1||W_2||W_3||\dots||W_{14}||W_{15}$  and every  $W_j$  consist of 32 bits
    - 3.2 for  $T=16$  to  $79$ .
      - 3.2.1  $W_T = (W_{T-3} \oplus W_{T-8} \oplus W_{T-14} \oplus W_{T-16}) \lll 1$ .
    - 3.3 for  $t=0$  to  $19$  .
      - 3.3.1  $f_t(B,C,D)=(B \wedge C) \vee ((\overline{B}) \wedge D)$ .
    - 3.4 for  $t=20$  to  $39$ .
      - 3.4.1  $f_t(B,C,D)=(B \oplus C \oplus D)$ .
    - 3.5 for  $t=40$  to  $59$ .
      - 3.5.1  $f_t(B,C,D)=(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$ .
    - 3.6 for  $t=60$  to  $79$ .
      - 3.6.1  $f_t(B,C,D)=(B \oplus C \oplus D)$ .
    - 3.7  $A=H_0$ ,  $B=H_1$ ,  $C=H_2$ ,  $D=H_3$ ,  $E=H_4$ .
    - 3.8 for  $t=0$  to  $79$ .
      - 3.8.1  $Temp=(A \lll 5) + f_t(B,C,D) + E + W_T + K_t$ .
      - 3.8.2  $E=D$ .
      - 3.8.3  $D=C$ .
      - 3.8.4  $C=(B \lll 30)$ .
      - 3.8.5  $B=A$ .
      - 3.8.6  $A=Temp$ .
    - 3.9  $H_0=H_0+A$ ,  $H_1=H_1+B$ ,  $H_2=H_2+C$ ,  $H_3=H_3+D$ ,  $H_4=H_4+E$ .
    - 3.10 Append  $H_0||H_1||H_3||H_4$ . The result 160 bits.
- 

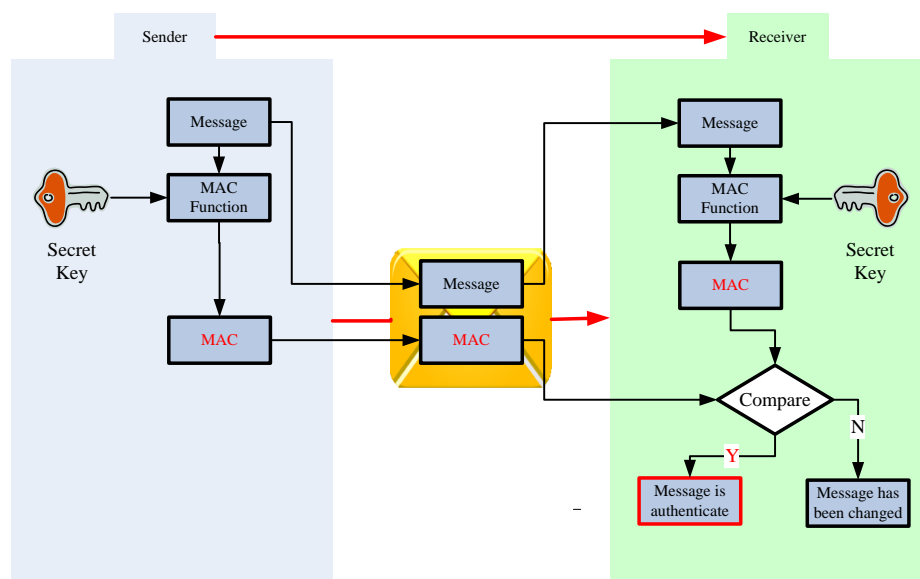
**Figure 13** The Secure Hash Algorithm-1 (SHA-1)

## 2.6 Message Authentication Code (MAC)

The MAC function is a type of symmetric key cryptography. It meets different security requirements, resists existential bogus plaintext attacks and confirms that the data has not been changed. Consequently, it does not allow malicious nodes to join the network through exchange messages within the network. In addition, it supports data authentication and integrity. However, the sent message often consists of two parts, as shown in Fig. 14. The first part is the plain text and the other part of the message is the cipher text via the shared key and is called MAC. At the end, the two parts of the message are appended and then sent as a message of a fixed size or both are sent separately in a sequence. At the node at the other side, when the received message takes the first part of the message, it is then encrypted by the shared key and



the result is compared to the second part of the received message. If the result is true, it means the message has not been altered and this ensures that the message comes from a legal node. Moreover, if the result of comparison is false, it means the message comes from malicious node and is therefore discarded. Finally, the normal MAC is not more secure because the attacker can eavesdrop the message and acquire the shared key by comparing and computing the first part of message, which is the plain text of the MAC. However, in our proposed scheme, the first part of the message is encrypted, which confuses the attacker and prevents the attacker from gaining the shared key [26-28].



**Figure 14** The MAC

## 2.7 Elliptic Curve Cryptography (ECC)

The elliptic curve has a wealthy and interesting history after mathematicians studied the elliptic curve hundreds of years ago. It has been used to solve a variety of problems in different fields and has an important role in central areas of mathematics, especially in arithmetic algebraic geometry [39]. Its history harkens back to ancient Greece, when the ancient Greeks studied the characteristic of elliptic curve as an object in geometry and algebra. Moreover, some of the theory originated in the nineteenth century and was first studied in the nineteenth century by Abel, Jacobi, Legendre and Weierstrass. Curiously, James Thomson was the earliest person

to use the term “elliptic curve” in the literature in 1727 in “A Poem sacred to the Memory of Sir Isaac Newton” [40].

Neal Koblitz (Washington University) and Victor Miller (IBM Company) independently proposed using elliptic curve groups to design cryptographic systems that depend on the difficulty of Discrete Logarithm Problem on these groups in 1985 [41]. Elliptic Curve Cryptography (ECC) is a type of public key cryptography. In public key cryptography, each part (user, device and node) needs pair keys, one of which is the public key and the other the secret or private key in order to communicate. The public key of any user is distributed to all users for communication with each other in different ways. Furthermore, the private key of any user is secret, where only the user knows the key. Conversely, symmetric key cryptography has one key, as shown in Fig. 5 [42]. The attraction to Elliptic Curve Cryptography compared with RSA is that ECC offers the same level of security with smaller key sizes than RSA, as shown in Tab. 1. The ECC private key is faster than the RSA private key because it is small [43], Elliptic Curve Cryptography is a one-way function due to the Discrete Logarithm Problem [44].

**Table 1** NIST Recommended Key Sizes

<b>RSA key size Based on No. of (bits)</b>	<b>ECC key size Based on No. of (bits)</b>
1024	160
2048	224
3072	256

The main operation in ECC, which takes more than 80 percent of its execution time, is called scalar multiplication [45]. Eq. (2.1) is the ECC equation.

$$y^2 \pmod{p} = x^2 + ax + b \pmod{p} \quad (2.1)$$

Where  $p$ ,  $a$  and  $b$  are parameters  $\in F_p$ .

$$4a^3 + 27b \pmod{p} \neq 0 \quad (2.2)$$

And  $p$  is prime number. Eq. (2.2) validates the EC parameter. To understand Elliptic Curve Cryptography very well, we must know some basic mathematical terminology of ECC through the following:

### 2.7.1 Groups

The abstract mathematical object  $(G, \times)$  consists of set  $G$  with binary operation  $(\times)$ . Moreover, a set  $G$ , called elements of the group and the elements for example  $\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$  could be finite or infinite. Additionally, the binary operation consists of two operations one of which is called an additive group denoted by  $(+)$  and the other operation called a multiplicative group denoted by  $(\times)$ . In addition, the additive identity element is offset by 0, for example  $(9+0 = 9 = 0+9)$  and the additive inverse of  $a$  is offset by  $-a$ ; for example  $(9+(-9)) = 0 = ((-9)+9)$ . Furthermore, the multiplicative identity element is usually offset by 1. For example, if we have  $G_5 = \{0, 1, 2, 3, 4\}$ , then  $(3 \times 1 = 3 = 1 \times 3)$  and the multiplicative inverse of  $a$  is offset by  $a^{-1}$ . For example,  $((3 \times 3^{-1}) \bmod 5 = 3 \times 2 \bmod 5 = 6 \bmod 5 = 1)$ . Also, the binary operation  $(\times): G \times G \rightarrow G$  satisfies the following [39-46-47]:

- Closure: If  $a$  and  $b \in G$ , then  $a \times b$  is also  $\in G$ .
- Associativity mean for  $a, b, c \in G: (a \times b) \times c = a \times (b \times c)$ .
- Identity mean the exists  $e \in G$  such that for all  $a \in G: a \times e = a = e \times a$ .
- Inverse mean for all  $a \in G$ , there exists  $b \in G$  such that  $(a \times b) = e = (b \times a)$ .
- Commutative mean  $a \times b = b \times a$  for all  $a, b \in G$ .

#### 2.7.1.1 Finite group

There are two types of field in cryptography applications, one of which is the prime field  $F_p$  where  $P$  is prime number and the type is a binary field  $F_p = 2^m$  where  $m$  is a large integer number. Furthermore, the group is called a finite group if the  $G$  is a finite set and the number of elements in  $G$ , is called the order of  $G$ . For example, we assume  $p$  is a prime number and  $F_p = \{0, 1, 2, 3, 4, \dots, p-2, p-1\}$  refers to the set of integers modulo  $P$ . In addition, the additive in the finite group is denoted by  $(F_p, +)$ , the multiplicative in the finite group is denoted by  $(F_p, \times)$  and the triple  $(F_p, +, \times)$  is a finite field referred to more clearly as  $F_p$  [39-46-47].

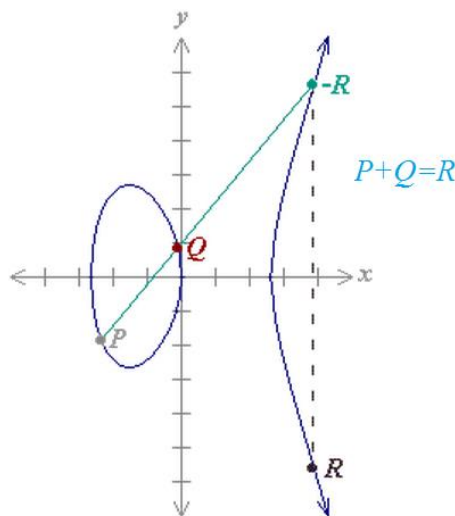
- Additive (+):  $F_p + F_p \rightarrow F_p$ , which is represent an abelian group.
- Multiplicative ( $\times$ ):  $F_p \times F_p \rightarrow F_p$ . which is represent an abelian group without zero ( $\setminus \{0\}$ ).
- Distribution:  $(a+b) \times c = (a \times c) + (b \times c)$  where  $a, b$  and  $c \in F_p$ .

### 2.7.1.2 Cyclic groups

The group can be cycle if there exists an item  $g$  in  $G$  such that every element  $x$  of  $G$  is an integral power of  $g$ , that  $x$  is of the form  $g^n$  for integer  $n$ . Moreover, the  $g$  is called a generator of the group. We can express the integral power of  $x$  in order to each item  $x$  in  $G$  like:  $x^0 = e$ ,  $x^1 = x$ ,  $x^2 = x * x$ ,  $x^3 = x * x * x$ . A cyclic group  $G$  generated by  $g$  is represented by  $\langle g \rangle$ . Finally, always it is abelian, and may be finite or infinite [47].

### 2.7.2 Adding point in ECC

Point addition (PA) is the process of adding the EC point  $P$  to another EC point  $Q$  with condition that  $P$  is not  $Q$  in order to find a third EC point  $R$  according to the operation ( $R = P + Q$ ). Then,  $R$  is a point located on the elliptic curve and it comes from reflection of  $-R$  over the  $x$ -axis, which intersects with the elliptic curve through  $P$  and  $Q$ , as shown in Fig. 15, and is a point in the elliptic curve group. It is not normal adding.



**Figure 15** Adding in ECC

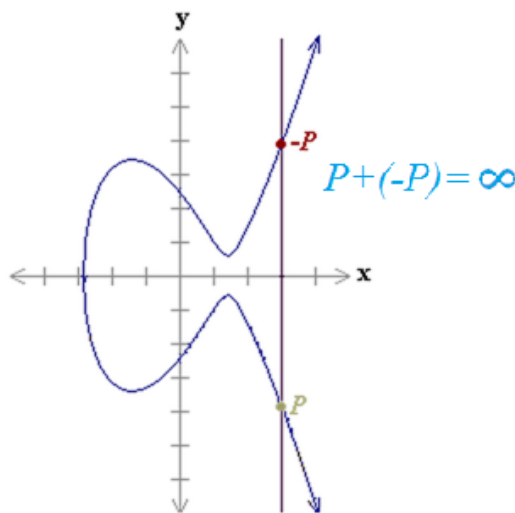
Moreover, it is done by special rules as following Eq. (2.3), (2.4) and (2.5) for finding the new point  $(x_R, y_R)$ :

$$\lambda = \frac{y_p - y_Q}{x_p - x_Q} \quad (2.3)$$

$$x_R = \lambda^2 - x_p - x_Q \quad (2.4)$$

$$y_R = -y_p + \lambda(x_p - x_R) \quad (2.5)$$

Furthermore, when adding point  $P(x, y)$  with  $-P(x, -y)$ , the result is not intersected with the elliptic curve at any point; thus, the result is  $(\infty)$  and acts as the additive identity, as shown in Fig. 16.



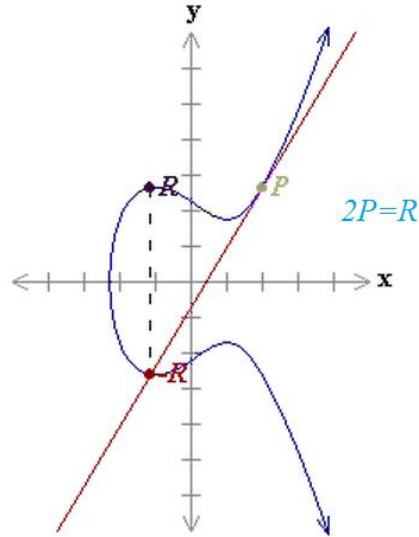
**Figure 16** Adding with  $P$  equal  $-P$  in ECC

In addition, the tangent line within  $P$  and  $-P$  is vertical and does not intersect the elliptic curve at any point to get a third point  $-R$ . Hence, the points  $P$  and  $-P$  cannot be added as previously because  $(P + (-P) = \infty)$  [48-49].

### 2.7.3 Doubling point in ECC

Point doubling (PD) is the process of adding the EC point  $P$  to itself with condition  $(y_p \neq 0)$  in order to find a third EC point  $R$  according to this operation  $(R = P + P)$ ;

then  $R$  is a point located on the elliptic curve and it comes from the reflection of  $-R$  over the  $x$ -axis, which intersects with elliptic curve through  $P$ , and it is a point in the elliptic curve group. Moreover, it is done by the operation ( $R=2\times P$ ) and the result of  $-R$ , which intersects with the elliptic curve through  $P$  is a reflection of  $-R$  over the  $x$ -axis, as shown in Fig. 17.



**Figure 17** Doubling point in ECC

Moreover, it is not normal multiplying. It is carried out by special rules as the following Eq. (2.6), (2.7) and (2.8) for finding the new point  $(x_R, y_R)$ :

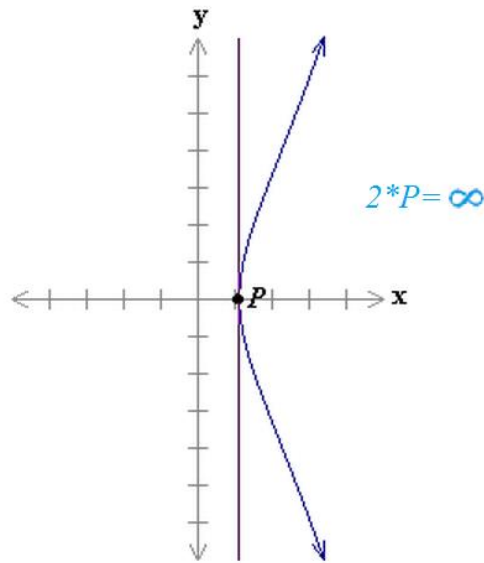
$$\lambda = \frac{3x_p^2 + a}{2y_p} \quad (2.6)$$

$$x_R = \lambda^2 - 2x_p \quad (2.7)$$

$$y_R = -x_p + \lambda(x_p - x_R) \quad (2.8)$$

Furthermore, when in adding point  $P(x, y)$  and  $y_P=0$  means  $P(x, 0)$ , the result is not intersected with the elliptic curve at any point therefore the result is  $(\infty)$ , as shown in

Fig. 18. Additionally, the points  $P(x,0)$  cannot be doubled as previously because  $(2 \times P = \infty)$  [48-49].



**Figure 18** Doubling point in ECC when  $y_p=0$

#### 2.7.4 Scalar point multiplication in ECC

Scalar point multiplication (SM) is a chain of doubling and adding rules to obtain the result, for example  $3P = ((2P) + P)$ . The first process is  $(2P)$ , which then adds its results to  $(P)$  to obtain the final result, which is a point on the elliptic curve [48-49].

#### 2.7.5 ECDLP

The resistance of the Elliptic Curve Discrete Logarithm Problem (ECDLP) is critical for the security of all elliptic curve cryptographic schemes. The ECDLP definition is given an Elliptic Curve  $E$  defined over  $F_p$ , a point (base point)  $P \in E_p(a,b)$  where  $p$  is an integer prime number and  $a,b$  is the EC domain parameter of order  $n$ , and a point (public key)  $Q \in \langle P \rangle$ , find the integer  $l \in [0, n - 1]$  such that  $Q = l \times P$  when  $l$  is the private key and an integer less than  $p$ . Therefore,  $l$  is called the discrete logarithm problem of the public key  $Q$  to base point  $P$ , symbolized as  $l = \log_p Q$ . Additionally, it is very difficult to determine the  $l$ . For example, let  $E_{29}(-1,16)$  and base point  $P(5,7)$ ,  $l=11$ , then the equation is clarified as:  $y^2 \text{ mod } 29 = (x^3 - x + 16) \text{ mod } 29$ .

Furthermore, the  $Q = l \times P \rightarrow Q = 11 \times (5,7) \rightarrow Q = (14,22)$ . Moreover, the  $l$  is the ECDLP of the  $(14,22)$  to the  $(5,7)$  so to find  $l$ , we must do  $(2((2(2(5,7))) + (5,7))) + (5,7)$ . Finally, if the ECDLP  $l$  is too large to compute, the  $l$  requires more time and it makes computing infeasible [39-47-50].

## 2.8 Key Management Goals

Key management is important and acts as the core to guarantee the communication security of a WSN. Additionally, an interesting problem for the resource-constrained sensor nodes is how to establish efficient key management in a WSN [8]. When the nodes are scattered randomly in an uncontrolled area and also unattended by humans, because the messages are sent wirelessly, they are prone to various attacks. Furthermore, the key management pertains to how to encrypt the messages in a way to prevent an enemy from decrypting the messages. This is done by using intrusion detection to prevent any fabricated nodes from joining the network and also by using tamper resistance to keep the material keys of encryption and decryption secure when the nodes are captured [51]. The key management schemes are divided into three categories: namely *symmetric*, *asymmetric* and *hybrid*. Finally, our proposed scheme uses the advantages of symmetric and asymmetric key cryptography via the SHA-1 in HMAC for cluster formation based on the symmetric key and sending the generated private key by [52] as asymmetric key cryptography, as illustrated in the next chapter in order to achieve data integrity, confidentiality and authentication, as well as to provide scalability and connectivity to the network.



## CHAPTER 3

### THE ENHANCED DATA SEQUENCE METHOD FOR ECC CRYPTOSYSTEM (EDSM-ECC)

#### 3.1 Introduction

Recently, a large number of studies has shown that public key cryptography, especially ECC, has many advantages in wireless communication due to the small size of the keys relative to other public key cryptography algorithms, such as RSA, and the mathematic difficulty to break and expose the key due the discrete logarithm problem (DLP) since RSA relies on the intractability of the integer factorization problem [53]. The proposed method develops and enhances the previous method which is used to secure the output of ECC by decreasing the size of the ECC point to 6 bits. In this chapter, we enhance the proposal in [25] by depending on the base of 6 when converting each digit to the sequence rather than the base of 2, and finally driving to reduce the number of bits. Therefore, the implementation of this study, which is based on the ECC under the finite field  $GF(p)$  with the idea of data sequence, provides better performance in terms of energy fatiguing, memory size and the time of processing when comparing it to other methods. Elliptic curve cryptography has many benefits in security and authentication algorithms with best performance, such as high security, high speed and low bandwidth requirements [54]. We will embed the data sequence with ECC in order to strengthen the ECC. In [25] the authors exploit the method of data sequencing to make the ECC more robust. The proposed method depends on [25] and developed its message size to be 6 bits rather than 8 bits. Subsequently, the size of the broadcasting message is eliminated. We apply this method of robust key management in hierarchical WSNs to generate private keys which are then distributed to each high sensor which is *CH* via the *BS* in addition to generating private keys, which are then distributed to cluster members such as the low sensors *L* to support security of the *CHs*.

### 3.2 Background on Data Sequence

In computer science, the data sequence is a fundamental data type and it is used widely in cryptography systems. Examples of data sequences include text files, stacks, queues and even strings represented as sequences or streams of characters or integer numbers, real numbers or binary numbers, etc. Furthermore, they are collections of objects ordered with homogeneous properties under any type and are countable. The difference between an array and a sequence is such that they are distinguished by special names for the two ends. A sequence can be empty of any object or contain one object or more, such as [ ] or [a] or [a,b,c] or [a<sub>0</sub>,....., a<sub>n</sub>]. Furthermore, no specific symbols are used to represent successive boundaries, which are written in rows such as (a<sub>1</sub>, a<sub>2</sub>, a<sub>3</sub>, ...), or as formulas such as (a<sub>0</sub>, a<sub>1</sub>, a<sub>2</sub>, ...), or (a<sub>0</sub>, a<sub>2</sub>, a<sub>4</sub>, ...). The symbols show the order of the border in the sequence; the first object in the sequence is a<sub>1</sub>, and second object is a<sub>2</sub>, third object is a<sub>3</sub> and so on, such as a prime sequence, ascending sequence, descending sequence, odd numbers, even numbers, natural number sequences, Hofstadter sequences or the Fibonacci sequence. The last example (the Fibonacci sequence) is formed through the first two numbers of the sequence being 1 and the third number from the result of summation of the first two numbers which are 1+1 =2. The fourth number is the summation of second and third numbers in the sequence: 1 + 2 = 3. The fifth number is the summation of the third and fourth numbers: 2 + 3 = 5, and so on. Therefore, the first ten numbers of the sequence are: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, as shown in Fig. 19 [55].

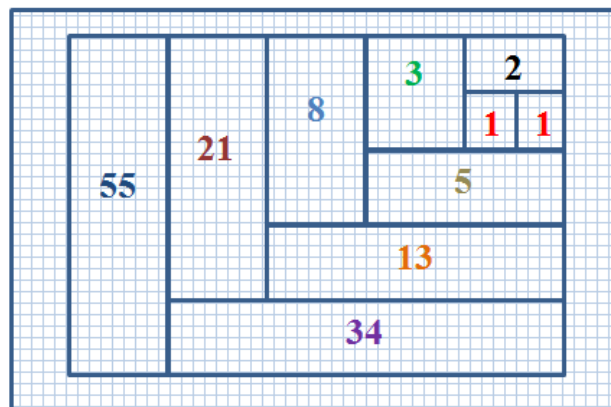


Figure 19 Fibonacci sequence

### 3.3 The Proposed Method

The proposed method relies on the idea of embedding the data sequence over the output of ECC to generate a series of vectors called  $S_i$  to give a series of bits of fixed length such as the Hash function, consequently confusing the adversary [25]. The propose method encompasses a number of procedures, as follows:

#### Procedure 1. Generating the sequence

To gain the Tab. 2 which is a data sequence, it must do the following steps:

- a) Assume  $P$  is a point generator and  $n$  is the order of  $P$ .
- b) Assume the sequence for  $i=0$  to  $n$  values.
- c) Convert every digit of the sequence in base 6 (this is differs from [25], which is in base 3).
- d) Assume  $m$  to be the number of digits such as:  $n=30$  therefore we find  $m=2$  (this differs from [25] in which  $m=4$ ) as shown in Tab. 2.

**Table 2** The Data Sequence Example

$n$	Sequence of $n=30$ & $m=2$		Corres-ponding	$n$	Sequence of $n=30$ & $m=2$		Corres-ponding
	Digit 1	Digit 0			Digit 1	Digit 0	
1	0	0	0	16	2	3	15
2	0	1	1	17	2	4	16
3	0	2	2	18	2	5	17
4	0	3	3	19	3	0	18
5	0	4	4	20	3	1	19
6	0	5	5	21	3	2	20
7	1	0	6	22	3	3	21
8	1	1	7	23	3	4	22
9	1	2	8	24	3	5	23
10	1	3	9	25	4	0	24
11	1	4	10	26	4	1	25
12	1	5	11	27	4	2	26
13	2	0	12	28	4	3	27
14	2	1	13	29	4	4	28
15	2	2	14	30	4	5	29

For instance, the verification of how the sequence 45 is equal to 29 can be explained as follows:

$45=(5*6^0)+(4*6^1)=5+24 \rightarrow n=29$ . While, in [25]  $1002=(2*3^0)+(0*3^1)+(0*3^2 )+(1*3^3)=2+0+0+27=29$ .

- e) Mapping the Tab. 2 into the following matrix: (this matrix differs from [25] in the number of columns, which is 4 columns).

$$M = \begin{bmatrix} a_{0,0} & a_{0,m-1} \\ a_{1,0} & a_{1,m-1} \\ \cdot & \cdot \\ \cdot & \cdot \\ a_{n,0} & a_{n,m-1} \end{bmatrix}$$

This matrix can be clarified more in the Eq. (3.1):

$$M = (n+1) \times m \quad (3.1)$$

- f) Make a circular right shifting for each row of  $M$  by one digit.

For example:  $[a_{i,0} \ a_{n,m-1}] \rightarrow [a_{n,m-1} \ a_{i,0}]$ .

(this sequence differs from [25] in the number of columns, which is 4 columns). As a result, the sequences formed will be follows:

$S: [S_0=[a_{0,m-1} \ a_{0,0}], S_1=[a_{1,m-1} \ a_{1,0}], \dots, S_n=[a_{n,m-1} \ a_{n,0}] ]$ .

## Procedure 2. Description of applying the generated sequence on the ECC

In this step, the procedure is explained in step1 and is applied to the ECC as in [25]. However, it is applied according to the differences that the proposed method achieves as follows:

- 1- Encrypting the message transferal

We assume that we have some elliptic curve  $E$  over a finite field  $GF(p)$  and that  $E$  and a point  $P \in E$  are known publicly, and the merge system  $m \rightarrow P_m$ ; which merges the clear text on an elliptic curve  $E$ . After that, when Alice decides to make a secret communication with Bob [25], they must take the following steps:

- a) Bob must select a random integer  $a$ , and propagate the point  $aP$  that will be a public key (where  $a$  remains secret and unknown to anyone but Bob) like in [25].

- b) Afterwards as in [25], Alice selects her own random integer  $l$  (where  $l$  is the private key) and computes two points:  $P_l$  is her public key as shown in Eq. (3.2), and  $P_2$  is the cipher text in the form of  $P_i$  (where  $P_i$  is a character) which will represent a point in  $E$  as shown in Eq. (3.3).

$$P_1(x_1 + y_1) = l \times p \quad (3.2)$$

$$P_2(x_2 + y_2) = P_i + l(a \times p) \quad (3.3)$$

- c) Compute  $S(x_1, y_1)$  and  $S(x_2, y_2)$  in which  $S$  is a corresponding sequence value obtained from Procedure 1\_(d,e) after applying the circular right shifting by one digit. Hence, the cipher message takes the form in Eq. (3.4).

$$C_m = (S(x_1, y_1), S(x_2, y_2)) \quad (3.4)$$

$C_m$  will be 4 digits such as 00 01, while in [25] it is 8 digits such as 0000 0001.

- d) Alice converts  $C_m$  to octal-binary representing a form (three bits for each digit) such as: 0 0  $\rightarrow$  000 000, 0 1  $\rightarrow$  000 001, 2 3  $\rightarrow$  010 111, 4 5  $\rightarrow$  100 101. However, in [25], 00 00 is converted to 0000 0000, 00 01  $\rightarrow$  0000 0001, 00 02  $\rightarrow$  0000 0010, etc. Finally, Alice will send the series of bits to Bob.

## 2- Decrypting the received message

To snatch the plain text from the cipher text, Bob knows the sequence of  $S_i$ , his own private key  $a$ , and the values of ECC parameters (base point  $P, a, b, p$ ). He receives the encrypted message as a series of bits as in [25] and do the following:

- a) Bob converts a binary bits form to the digits (binary  $\rightarrow$  digit), for examples: 000 000  $\rightarrow$  0 0, 000 001  $\rightarrow$  0 1, 010 111  $\rightarrow$  2 7, 100 101  $\rightarrow$  4 5. While in [25] Bob receives 16 bits and converted it to 8 digits as follows: 0000000000000001 be 0000 0001.
- b) Bob converts  $C_m$  into group of  $3m$  (digits), but in [25] is converted into  $2m$ .

- c) Bob gets a group of  $m$  digits from the sequence in the (b).
- d) Bob makes a circular left shifting to the sequence of  $m$  digits by one for two digits. But in [25] this shifting is for 4 digits.
- e) Transform the sequence digits to decimal form, and save the value in  $k$ .

For example:

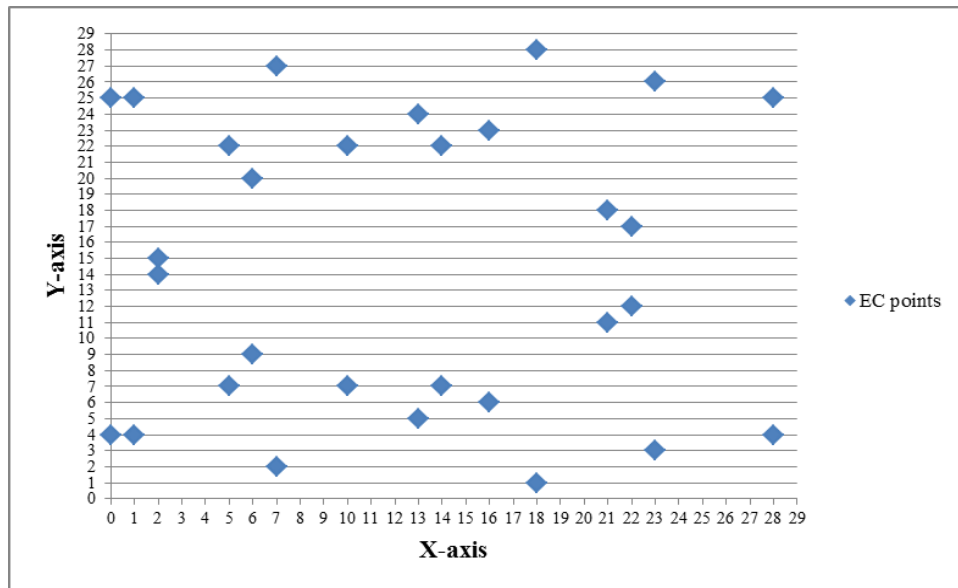
(011 001) is separated as: (011 = 3) and (001=1). Then it can be represented as (31) and be in the form of the following:

$$(1*6^0)+(3*6^1) = 1+18=19 \rightarrow k=19.$$

- f) Find a point  $(x_1, y_1)$  from  $(k+1)$ .
- g) Repeat the same steps explained in decryption part for the next element of the sequence of step (c) for the recovery of  $S(x_2, y_2)$ .
- h) In order to obtain  $P_i$  from  $P_i + l(aP)$ , Bob used his private key  $a$  and compute  $a(lP)$  from the first part of the pair, then subtracts it from the second part to find:  $P_i + l(aP) - a(lP) = P_i + laP - laP = P_i$ , and then separate the merging to get back the clear text as in [25].

### 3.4 Implementation of the Proposed Method

The proposed method is depending on the following Eq. (3.5) of the ECC as shown in Fig. 20.



**Figure 20** The points of the elliptic curve  $E_{29}(-1,16)$

$$Y^2 = X^3 - X + 16 \pmod{29} \quad (3.5)$$

And the chosen base point  $P$  is picked as  $(5, 7)$ . The set of all points in the curve is depicted in Tab. 3.

**Table 3** The Generation of the EC Points for  $E_{29}(-1,16)$

#	EC point	Equivalent alphabet	#	EC point	Equivalent alphabet
1P	(5,7)	a	16P	(0,25)	p
2P	(28,4)	b	17P	(1,25)	q
3P	(18,1)	c	18P	(7,2)	r
4P	(22,12)	d	19P	(16,6)	s
5P	(6,20)	e	20P	(14,7)	t
6P	(13,5)	f	21P	(10,22)	u
7P	(2,14)	g	22P	(23,26)	v
8P	(21,11)	h	23P	(21,18)	w
9P	(23,3)	i	24P	(2,15)	x
10P	(10,7)	j	25P	(13,24)	y
11P	(14,22)	k	26P	(6,9)	z
12P	(16,23)	l	27P	(22,17)	,
13P	(7,27)	m	28P	(18,28)	.
14P	(1,4)	n	29P	(28,25)	/
15P	(0,4)	o	30P	(5,22)	

The number of generated points for the Eq. (3.5) is 30 points. 26 of them are given to the English alphabetic, and the remaining points are given to some other special characters (‘,’ , ‘.’ , ‘/’ , and ‘space’). Recall that the number of points can be increased to more characters in the case of changing the EC Eq. (3.5) as in [25]. For example, suppose that Alice wants to encrypt and send a message "hello" to Bob. Thus, Alice must follow the following steps:

Step1. Establishing the data sequence:

- $P$  is a point generator with order  $n=30$ , and  $m=2$ .

- Convert a sequence 0 to  $n$  to the form similar to Tab. 2, and it can be represented as a matrix named  $M$  [30×2].
- Circularly right shifting each row of  $M$  by one digit yields as shown in Tab. 4, and get  $M_s$  where  $s$  mean shift:

**Table 4** The Data Sequence After Circularly Right Shifting

$n$	Sequence of $n=30$ & $m=2$		Corresponding	$n$	Sequence of $n=30$ & $m=2$		Corresponding
	Digit 1	Digit 0			Digit 1	Digit 0	
1	0	0	0	16	3	2	15
2	1	0	1	17	4	2	16
3	2	0	2	18	5	2	17
4	3	0	3	19	0	3	18
5	4	0	4	20	1	3	19
6	5	0	5	21	2	3	20
7	0	1	6	22	3	3	21
8	1	1	7	23	4	3	22
9	2	1	8	24	5	3	23
10	3	1	9	25	0	4	24
11	4	1	10	26	1	4	25
12	5	1	11	27	2	4	26
13	0	2	12	28	3	4	27
14	1	2	13	29	4	4	28
15	2	2	14	30	5	4	29

- The sequence formed will be: [00], [10], [20], [30], [40], [50], [01], [11], [21], [31], [41],[51], [02], [12], [22], [32], [42], [52], [03], [13], [23], [33],[43], [53], [04], [14], [24], [34], [44], [45].

Step2. Encryption mechanism:

The encryption mechanism can be described and clarified through the following example according to Fig. 21:



- 
1. READ: EC domain parameters  $(p, a, b, P, n)$ . Where  $p$  is prime of the EC equation,  $a$  and  $b$  is integer numbers,  $P$  is base point and  $n$  is the group order.
  2. Select  $l \in F_p [1, n-1]$ . Where  $F_p$  is finite group and  $l$  is the private key of receiver.
  3. Read: *cipher message*. Which is 12 bits.
  4. For  $i=1$  to 6.
    - 4.1.  $X[i] = \text{cipher message}[i]$ .
  5.  $X = \text{Convert } X1 \text{ to data sequence}$ .
  6. Left shift by 1 to  $X$ .
  7.  $K_X = (\text{digit}_1 * \delta^0) + \dots + (\text{digit}_n * \delta^n)$ .
  8.  $N_X = K_X + l$ .
  9. Calculate  $X_x = N \times P$ .
  10. For  $i=7$  to 12.
    - 10.1.  $Y1 = \text{cipher message}[i]$ .
  11.  $Y = \text{Convert } Y1 \text{ to data sequence}$ .
  12. Left shift by 1 to  $Y$ .
  13.  $K_Y = (\text{digit}_1 * \delta^0) + \dots + (\text{digit}_n * \delta^n)$ .
  14.  $N_Y = K_Y + 1$ .
  15. Calculate  $Y_y = N_Y \times P$ .
  16. Calculate  $G = X * l$ .
  17. Calculate  $D = Y_y - G$ . Subtracting done by them location in table 3-3.
  18. Return (*plaint text* = convert  $D$  to character according to table 3-3).
- 

**Figure 21** Encryption by EDS-ECC algorithm

Initially, suppose that the private key ( $l$ ) of Alice is 13, and the private key ( $a$ ) of Bob is 24. The plaintext (message) is 'h'. Therefore, to encrypt 'h', Alice must attain the following steps:

- Compute the public key of Bob:
 
$$Q_B = aP = 24(5, 7) = (2, 15).$$
- Convert the plain text 'h' to the corresponding ECC point as in Tab. 3:
 
$$\text{Plain text ('h')} = (21, 11).$$
- Calculate  $lP_B$  :
 
$$lP_B = 13(2, 15) = (28, 4).$$
- Encrypting 'h':
 
$$\text{Cipher message} = \text{plain text } (P_i) + l (Q_B)$$

$$\begin{aligned}
&=(21,11)+(28,4) \\
&=(10,7).
\end{aligned}$$

- Calculate public key of Alice ( $Q_A$ ):  
 $1P=13(5,7)=(7,27)$ .
- Encrypted version of the message is:  $C_m=(S(7,27),S(10,7))$ ,  
where:  $x_1=7, y_1=27, x_2=10$  and  $y_2=7$ .
- Alice attains Procedure\_1 to generate the sequence  $S$ ,  
where,  $S(7,27)=20$  and  $S(10,7)=13$ .
- Notice that the message is converted into “2013”.
- Circularly right shifting for each  $S$ : “0231”.
- Finally, convert the transmitted message “0231” to a stream of bits: “**000 010 011 001**”.

Step3. Decrypting mechanism:

After receiving the encrypted message, Bob must attain the following steps to recover the plain text ‘h’ according Fig. 22:

- To decrypt the (000010), we convert it to digits (000=0) and (010=2).
- Bob gets two digits (02).
- Making a circular left shift to the sequence by one digit yields (20).
- Transform a sequence (20) to decimal form, and save the value in  $k$  as Eq. (3.6).

$$k = (\text{digit}_0 \times 6^0) + (\text{digit}_1 \times 6^1) + \dots + (\text{digit}_n \times 6^n) \quad (3.6)$$

- $k = (0 * 6^0) + (2 * 6^1) = 0 + 12 = 12$  as in Eq. (3.6).
- Find the point from a pre-computed and stored point  $(k+1)=(x_1, y_1)$ .
- $P=12+1=13$ .
- Therefore,  $(x_1, y_1)=13(5,7)=(7,27)$ . Where  $(7, 27)$  represents  $kP$  and is the public key for Alice. Similarly, to recover the other points, the data sequence in Procedure 1 is attained. Then, the encrypted version  $((7, 27), (10, 7))$  is recovered to extract  $P_i$ .

- Bob must compute his private key and Alice's public key as follows:

$$P_i = \text{Cipher message} - a * (IP)$$

$$= (10,7) - 24(7,27)$$

$$= (10,7) - (28,4)$$

$$= (21,11)$$

= 'h'. This acts as the plain text.

- 
1. READ: EC domain parameters  $(p, a, b, P, n)$ . Where  $p$  is prime of the EC equation,  $a$  and  $b$  is integer numbers,  $P$  is base point and  $n$  is the group order.
  2. Select  $l \in F_p [1, n-1]$ . Where  $F_p$  is finite group and  $l$  is the private key of receiver.
  3. Read: *cipher message*. Which is 12 bits.
  4. For  $i=1$  to 6.
    - 4.1.  $X[i] = \text{cipher message}[i]$ .
  5.  $X = \text{Convert } X1 \text{ to data sequence}$ .
  6. Left shift by 1 to  $X$ .
  7.  $K_X = (\text{digit}_1 * \delta^0) + \dots + (\text{digit}_n * \delta^n)$ .
  8.  $N_X = K_X + l$ .
  9. Calculate  $X_x = N_X * P$ .
  10. For  $i=7$  to 12.
    - 10.1.  $Y1 = \text{cipher message}[i]$ .
  11.  $Y = \text{Convert } Y1 \text{ to data sequence}$ .
  12. Left shift by 1 to  $Y$ .
  13.  $K_Y = (\text{digit}_1 * \delta^0) + \dots + (\text{digit}_n * \delta^n)$ .
  14.  $N_Y = K_Y + 1$ .
  15. Calculate  $Y_y = N_Y * P$ .
  16. Calculate  $G = X * l$ .
  17. Calculate  $D = Y_y - G$ . Subtracting done by them location in table 3-3.
  18. Return (*plaint text* = convert  $D$  to character according to table 3-3).
- 

**Figure 22** Decryption by EDS-ECC algorithm

### 3.5 Analysis of the Proposed Method

The proposed method provides a better performance in terms of memory size requirements in comparison with [25]. This reducing of memory usage can be easily depicted for both methods (proposed and [25]) through the following example:

To encrypt the "hello" message with the proposed method according to Tab. 5, the following stream of bits will be sent:

“000010011001000010000001000010001010000010001010000010001010000010100010”

while in [25] the stream of bits that will be sent is:

“00000101000001000000010100000010000001010100010100000101010001010001010000101000110” as shown in Tab. 6.

The Tab. 7 shows the difference between the proposed methods and [25] in terms of number of bits and digits for encrypting/decrypting the plain text “HELLO”.

The Tab. 8 and Fig. 23 displays the difference between the proposed methods and [25] in terms of number of bits for sending the plain text “HELLO”.

**Table 5** Encryptions in EDSM-ECC Method

Plain Text	Point $P_i$	Encryption before applying sequence $C_m = (lP, P_i + lQ_B)$	Encryption After applying sequence $C_m = (S(x_1, y_1), S(x_2, y_2))$	Convert the sequence to (12 bits) for sending
h	(21, 11)	((7, 27), (10, 7))	<b>02 31</b>	<b>000010</b> 011001
e	(6, 20)	((7, 27), (2, 14))	<b>02 01</b>	<b>000010</b> 000001
l	(16, 23)	((7, 27), (1, 4))	<b>02 12</b>	<b>000010</b> 001010
l	(16, 23)	((7, 27), (1, 4))	<b>02 12</b>	<b>000010</b> 001010
o	(0, 4)	((7, 27), (1, 25))	<b>02 42</b>	<b>000010</b> 100010

**Table 6** Encryptions in [25] Method

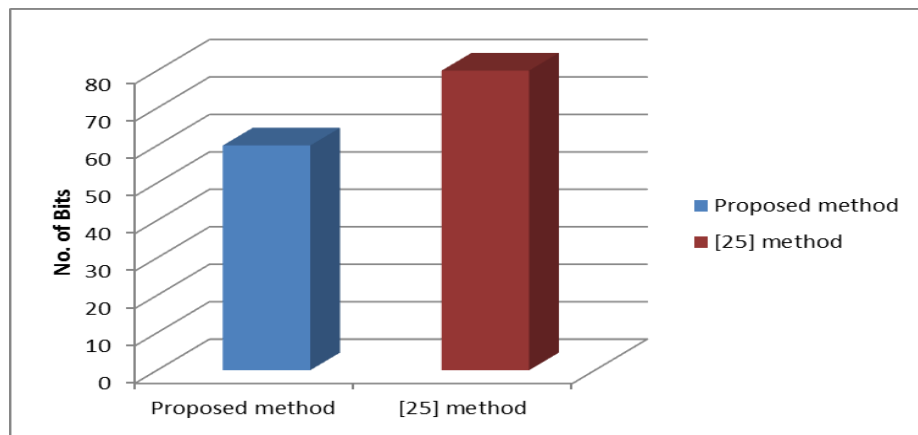
Plain Text	Point $P_i$	Encryption before applying sequence $C_m = (lP, P_i + lQ_B)$	Encryption After applying sequence $C_m = (S(x_1, y_1), S(x_2, y_2))$	Convert the sequence to (12 bits) for sending
h	(21, 11)	((7, 27), (10, 7))	<b>0011 0010</b>	<b>00000101</b> 00000100
e	(6, 20)	((7, 27), (2, 14))	<b>0011 0002</b>	<b>00000101</b> 00000010
l	(16, 23)	((7, 27), (1, 4))	<b>0011 1011</b>	<b>00000101</b> 01000101
l	(16, 23)	((7, 27), (1, 4))	<b>0011 1011</b>	<b>00000101</b> 01000101
o	(0, 4)	((7, 27), (1, 25))	<b>0011 1012</b>	<b>00000101</b> 01000110

**Table 7** A Comparison in Terms of no. of Bits and Digits

Plain Text	Encryption After applying data sequence $C_m = (S(x_1, y_1), S(x_2, y_2))$ (EDSM-ECC)		Encryption After applying data sequence $C_m = (S(x_1, y_1), S(x_2, y_2))$ (Method in[25])	
	h	0231	4-digits (12-bit)	00110010
e	0201	00110002		
l	0212	00111011		
l	0212	00111011		
o	0242	00111012		

**Table 8** Comparison in Terms of no. of Bits for Sending Hello

Plain text	No. of bits for sending message in EDSM-ECC	No. of bits for sending message in [25]
hello	$5 * 12 = 60$	$5 * 16 = 80$



**Figure 23** A comparison in terms of bits no. to send hello

### 3.6 Consequence

The EDSM-ECC developed the previous method [25] and uses the idea of a data sequence to protect the output of the ECC. This development concentrates on the reduction of the number of bits for the sent encrypted message. The proposed method (EDSM-ECC) proves that it is far smaller than the compared method in terms of number of bits. Moreover, we use this method in our proposed scheme to encrypt the transferred private key for the *CH* sensors and *L* sensors.

## CHAPTER 4

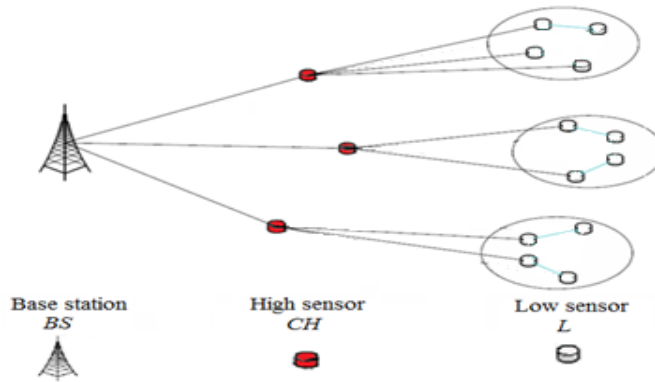
### SYSTEM IMPLEMENTATION AND RESULTS

#### 4.1 Introduction

A WSN consists of a large number of small, self-powered, and inexpensive devices that have ability to sense, compute, and communicate with each other through wireless techniques. Sensor nodes collect the sensing data from the environment then send collected data to the end user. In this chapter, a robust key management scheme for hierarchical WSNs based on ECC over a finite field and HMAC to provide secure links between the nodes within network and to form the clusters is implemented with the recorded results. The simulation results using the MATLAB program is depicted. The proposed scheme provides acceptable energy consumption, connectivity, security goals, and resistance upon threads compared with other schemes.

#### 4.2 The Network Model

The proposed method considered the hierarchical structure for the sensor network. The network consists of three levels: the top level comprises the *BS* with unlimited resources; the middle level consists of *CH* sensors which are high sensors working as cluster heads with properties higher and operating with far more power than *L* sensors, which are also the sensors, but working at the last level of a network. Moreover, the routing of gathered information in the proposed network is central and prepared in a secure way by using the techniques of cryptography from the *L* sensors to the *BS* via the *CH* sensors. Furthermore, *L* sensors can be connected to the *CH* sensors directly or through other *L* sensors. This means that the *L* sensors can be connected by one hop to the *CH* sensors or by multi hops. The *CH* sensors can directly connect to the *BS* within the network, as shown in Fig. 24.



**Figure 24** The hierarchical structure for the sensor network

### 4.3 Assumptions

- The *BS* is trusted with a large memory size, high power processing, powerful transmission range, and unlimited battery energy.
- The *BS* and each sensor are assigned with a unique *ID*.
- The *BS* has the *IDs* of all sensor nodes in a table.
- The deployment area is  $100 * 100 \text{ m}^2$ .
- Number of *CH* sensors are 2, which are equipped with tamper resistance and the number of *L* sensors are 100.
- The range of all *L* sensors nodes transmission is 30 m.
- All sensors nodes (*CH* and *L*) are static.
- *L* sensors are not equipped with tamper resistance so as to reduce cost.

### 4.4 The Network Phases

The proposed scheme, which is named as “A Robust Key management Scheme for hierarchical WSNs”, consists of three phases as depicted bellow:

#### 4.4.1 Pre-distribution phase

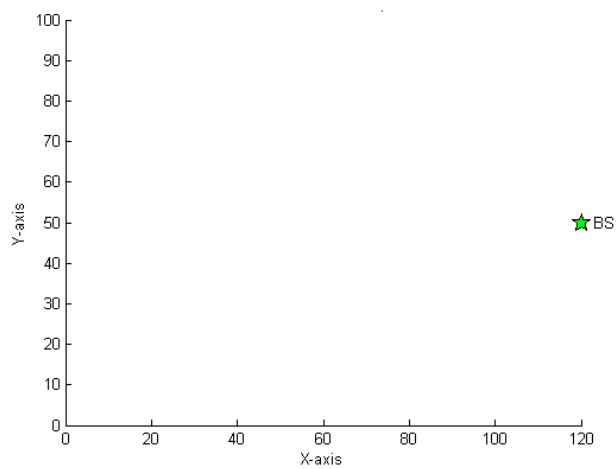
1. The *BS* is preloaded with  $ID_{BS}$ , *ID* of all *CH* sensors and *ID* of all *L* sensors. Moreover, it is preloaded with the ECC parameter, SHA-1, Tab. 2 and Tab. 3.

2. Each  $CH$  sensor is preloaded with its identity  $ID_{CH}$ ,  $ID_{BS}$ , ECC parameters (addition and multiplication), SHA-1, tamper resistance to protect the material, one seed key and Tab. 2.

3. Each  $L$  sensor is preloaded with its identity  $ID_L$ , the identity of all  $CH$  sensors ( ), ECC parameters (addition and multiplication), SHA-1, one seed key and Tab. 3.

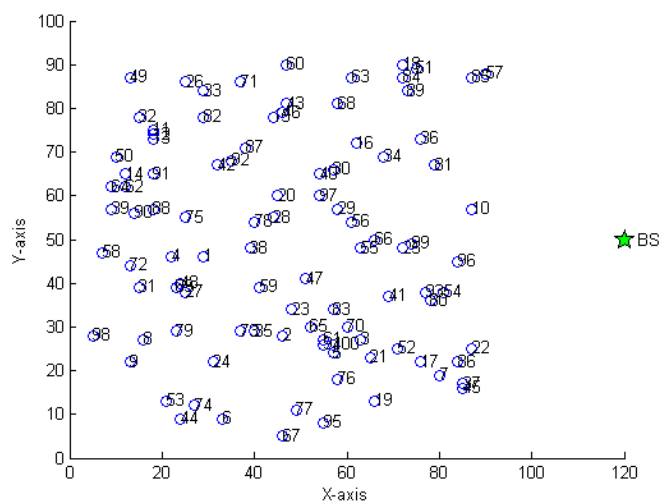
#### 4.4.2 Distribution phase

Due to the random scatter of the sensors, the network topology is unpredictable. The BS is deployed in a protected area, as shown in Fig. 25.



**Figure 25** The BS setup

Furthermore, 100  $L$  sensors are deployed within  $100*100m^2$  randomly, as shown in Fig. 26.

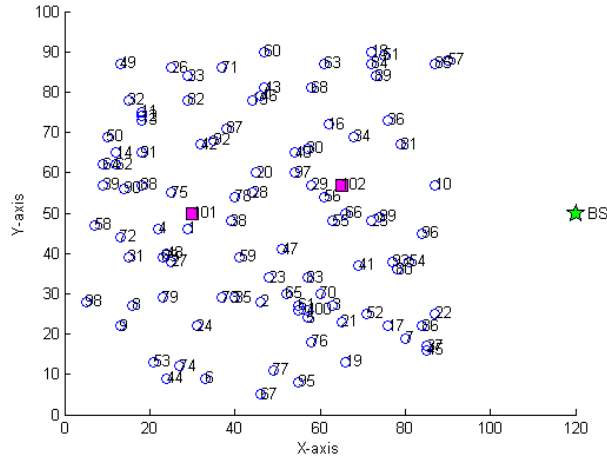


**Figure 26** The Ls setup



Farther, we are deploying two *CH* sensors within  $100 \times 100 \text{m}^2$  randomly, as shown in Fig. 27.

For exchanging the data between the nodes within the network, and create the secure link between the nodes, they are must follow the follows steps:

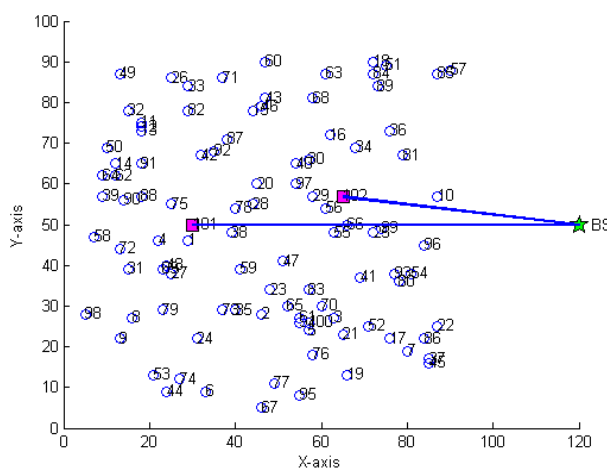


**Figure 27** The *CH*s setup

### Step 1. The Share key sent from *BS* to *CH*s

1.1 After the deployment, the *BS* generates a *Share\_Key* (it is a random symmetric key number) for each *CH* sensor as shown in Fig. 28 to prevent the attacker from getting the share key when physically captured from attacker. Moreover, it used a complex algorithm to encrypt the *Share\_Key* by its  $ID_{BS}$  as Eq. (4.1), Fig. 29 and Fig. 30.

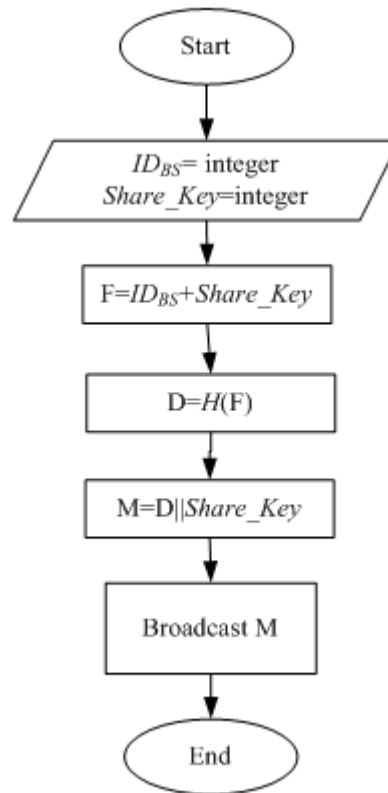
$$BS \rightarrow CH_i : H(ID_{BS} + Share\_Key) \parallel Share\_Key \quad (4.1)$$



**Figure 28** The sending of encrypted *Share\_Key* from *BS*

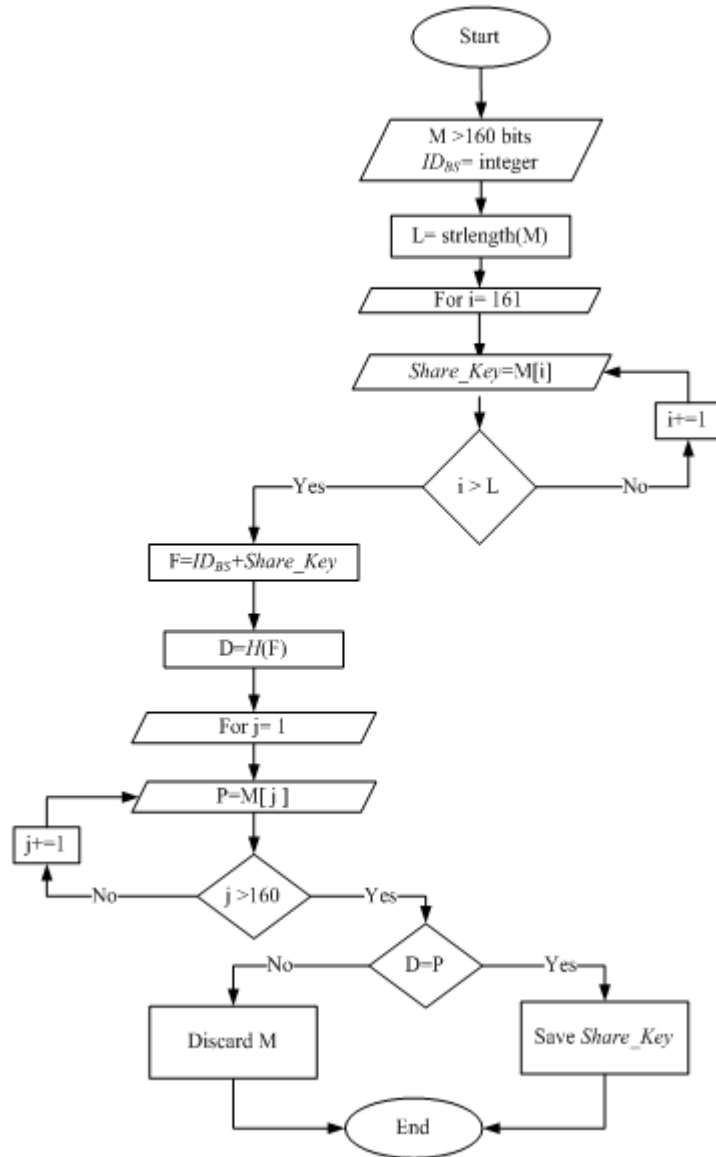
- 
1.  $ID_{BS}$  =integer,  $Share\_Key$ =integer.
  2.  $F=ID_{BS}+Share\_Key$ .
  3.  $D=H(F)$ .                      Where  $D=160$  bits and  $H$  is SHA-1.
  4.  $M=D||Share\_Key$ .      Where  $M$  is the message.
  5. Broadcast  $M$ .
  6. End.
- 

**Figure 29** The sending of encrypted  $Share\_Key$  from  $BS$  algorithm



**Figure 30** The sending of encrypted  $Share\_Key$  from  $BS$  flowchart

1.2 Each CH sensor receives the message  $M$  and only they can decrypt the  $M$  because only they have the  $ID_{BS}$ . Moreover, they decrypt the message by Fig. 31 and Fig. 32.



**Figure 31** The decryption of a *Share\_Key* by each *CH* flowchart

- 
1.  $ID_{BS}$  =integer.
  2.  $M > 160$ .                      Where it's more than 160 bits acts the received message.
  3. For  $i=161$  to  $strlength(M)$ .
    - 3.1.  $Share\_Key=M[i]$ .
  4.  $F=ID_{BS}+Share\_Key$ .
  5.  $D=H(F)$ .                      Where  $D=160$  bits and  $H$  is SHA-1.
  6. For  $j=1$  to 160.
    - 6.1.  $P=M[j]$ .
  7. If ( $D=P$ ).
    - 7.1. Save  $Share\_Key$ .
    - 7.2. End
  8. Else
    - 8.1. Discard  $M$ .
  9. End.
- 

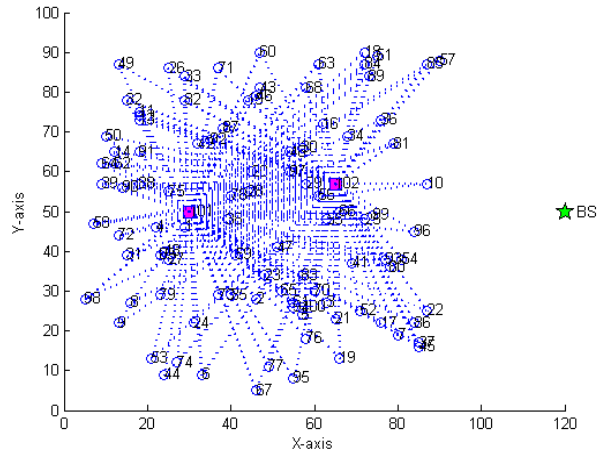
**Figure 32** The decryption of a  $Share\_Key$  by each  $CH$  algorithm

### Step 2. Cluster formation

The use of the clustering method makes the algorithm more economical than the flat sensor network because in the cluster model, only the  $CH$  sensors have the ability to analyze data and communicate with the  $BS$ . Therefore, the cluster method reduces overheads. However, in the flat model, each node can analyze data and communicate with the  $BS$ . Moreover, every node has the ability to communicate with the  $BS$ . Additionally, the cluster formation starts from the  $CH$  sensors and performs the following step:

- 2.1 Each  $CH$  sensor broadcasts a hello message [20], which is the  $ID_{CH}$  encrypted by SHA-1, which then broadcasts this message, as shown in Fig. 33 by Eq. (4.2) , Fig. 34 and Fig. 35.

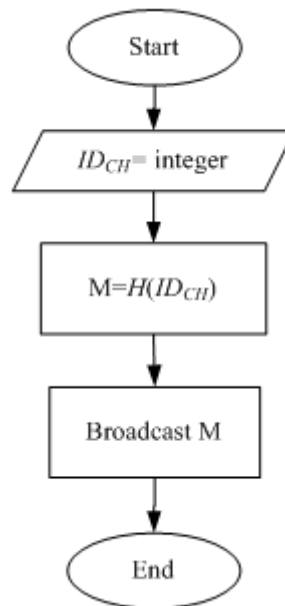
$$CH \rightarrow L: H(ID_{CH}) \quad (4.2)$$



**Figure 33** The CHs that broadcast Hello Message

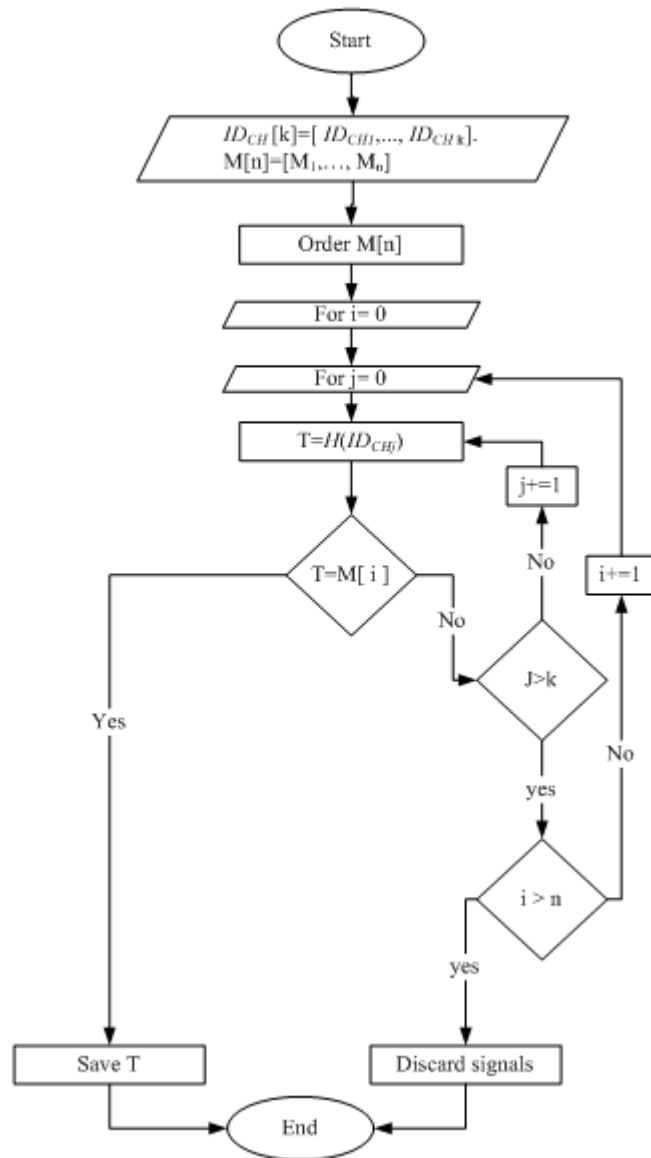
- 
1.  $ID_{CH}$ =integer.
  2.  $M=H(ID_{CH})$ .                      Where  $M=160$  bits and  $H$  is SHA-1.
  3. Broadcast  $M$ .
- 

**Figure 34** The CHs that broadcast Hello Message algorithm



**Figure 35** The CHs that broadcast Hello Message flowchart

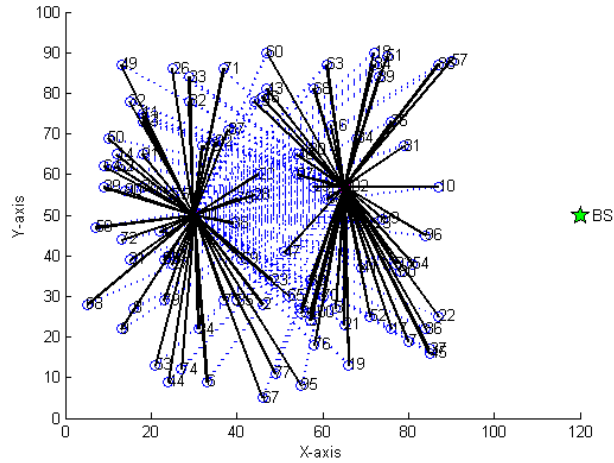
2.2 Each  $L$  sensor receives more than one hello message [20] as the signals and order these signals ascendingly, start from strongest to weakest according to flowchart in Fig. 36 and the algorithm in Fig. 37, and as shown in Fig. 38 . The  $L$  sensors can decrypt the messages because they preloaded with the  $ID$  of each  $CH$  sensor in pre-distribution phase.



**Figure 36** The CH selection by the  $L$  sensors flowchart

- 
1.  $ID_{CH}[k]=[ID_{CH1}, \dots, ID_{CHk}]$ .
  2.  $M[n]=[M_1, \dots, M_n]$ . Where  $M$  received messages.
  3. Order  $M[n]$ .
  4. For  $i=0$  to  $n$ .
    - 4.1. For  $j=0$  to  $k$ .
      - 4.1.1.  $T=H(ID_{CHj})$ . Where  $T=160$  bits and  $H$  is SHA-1.
      - 4.1.2. If  $(T=M[i])$ .
        - 4.1.2.1. Save  $T$ . Where  $T$  is the strongest signal.
        - 4.1.2.2. End.
  5. Out "Discard Messages"
  6. End.
- 

**Figure 37** The CH selection by the  $L$  sensors algorithm



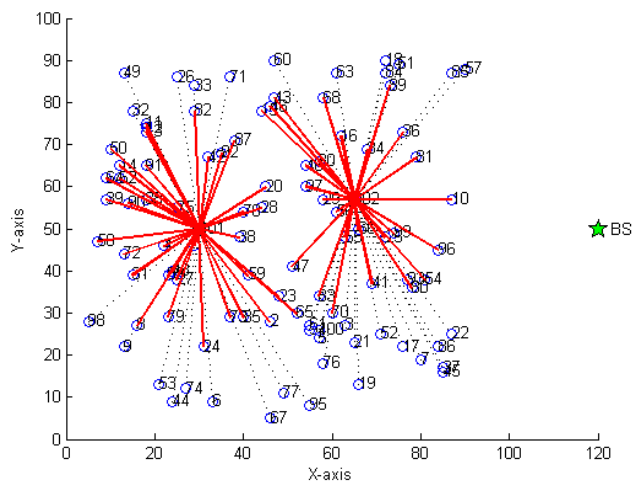
**Figure 38** The  $CH$  selection by the  $L$  sensors

2.3 Each  $L$  sensor sends  $ID_L$  encrypted by  $ID_{CH}$  to its  $CH$  via Eq. (4.3) , Fig. 39 and Fig. 41 after picked it, then broadcasts stream of bits and waiting the acknowledgement from selected  $CH$  as shown in Fig. 40. In addition the dotted line refers to the remote sensor which its message not reached to  $CH$  sensors.

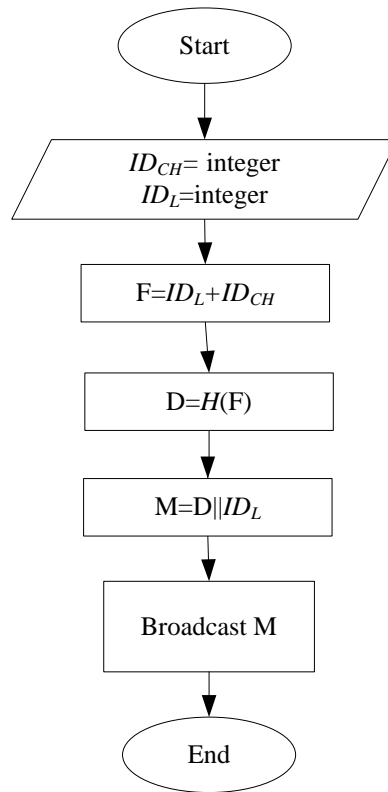
$$L \rightarrow CH_i : H(ID_L + ID_{CH}) \parallel ID_L \quad (4.3)$$

- 
1.  $ID_{CH}$  =integer,  $ID_L$  =integer.
  2.  $F = ID_L + ID_{CH}$ .
  3.  $D = H(F)$ .                      Where  $D=160$  bits and  $H$  is SHA-1.
  4.  $M = D \parallel ID_L$ .                Where  $M$  is the message.
  5. Broadcast  $M$ .
- 

**Figure 39** Each  $L$  sends its ID to selected  $CH$  algorithm



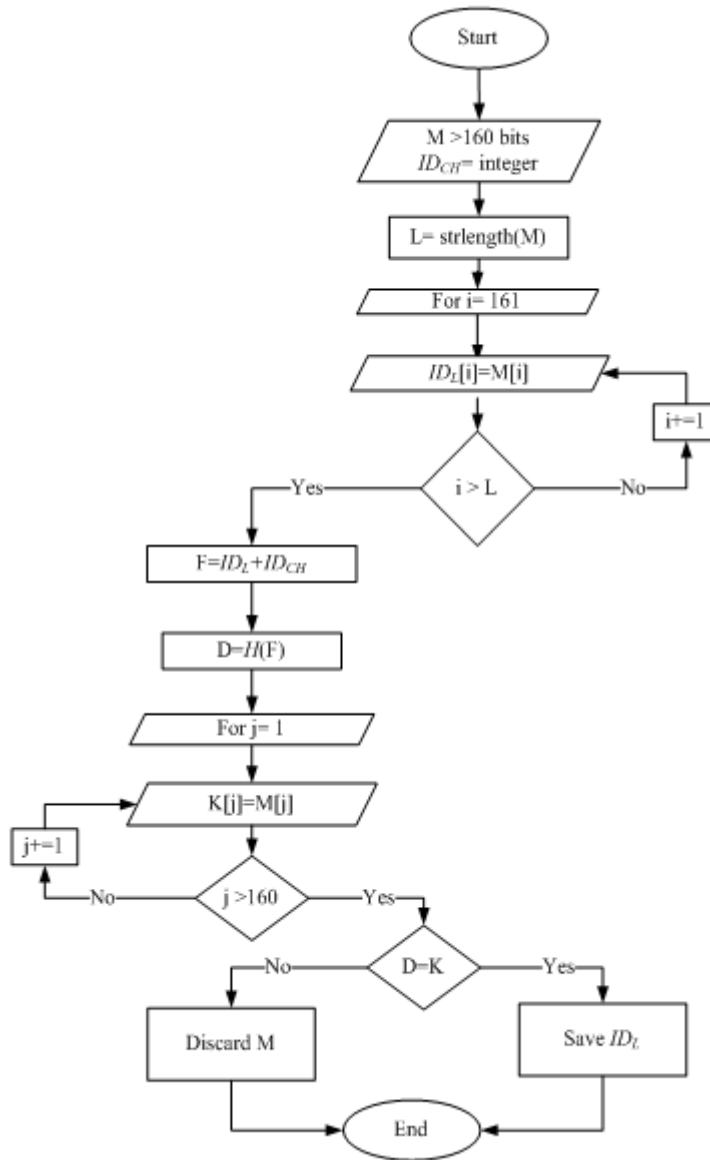
**Figure 40** Each  $L$  sends its ID to selected  $CH$



**Figure 41** Each  $L$  sends its ID to selected  $CH$  flowchart

2.4 Each  $CH$  sensor receives a request from the  $L$  sensors as shown in Fig. 40 which is a stream of bits to become a member within their cluster. The  $CH$  sensors can decrypt the message and gain the  $ID$  of their member ( $ID_L$ ) according to Fig. 42 and Fig. 43. Moreover, the  $CH$  sensors can recognize the legal node from malicious nodes.





**Figure 42** Decryption of  $L$  members messages by each  $CH$  flowchart

- 
1.  $ID_{CH}$  =integer.
  2.  $M > 160$ .      Where it's more than 160 bits acts the received message.
  3. For  $i=161$  to  $\text{strlength}(M)$ .
    - 3.1.  $ID_L = M[i]$ .
  4.  $F = ID_L + ID_{CH}$ .
  5.  $D = H(F)$ .      Where  $D=160$  bits and  $H$  is SHA-1.
  6. For  $j=1$  to 160.
    - 6.1.  $K[j] = M[j]$ .
  7. If ( $D=K$ ).
    - 7.1. Save  $ID_L$ .
    - 7.2. End.
  8. Else
    - 8.1. Discard  $M$ .
  9. End.
- 

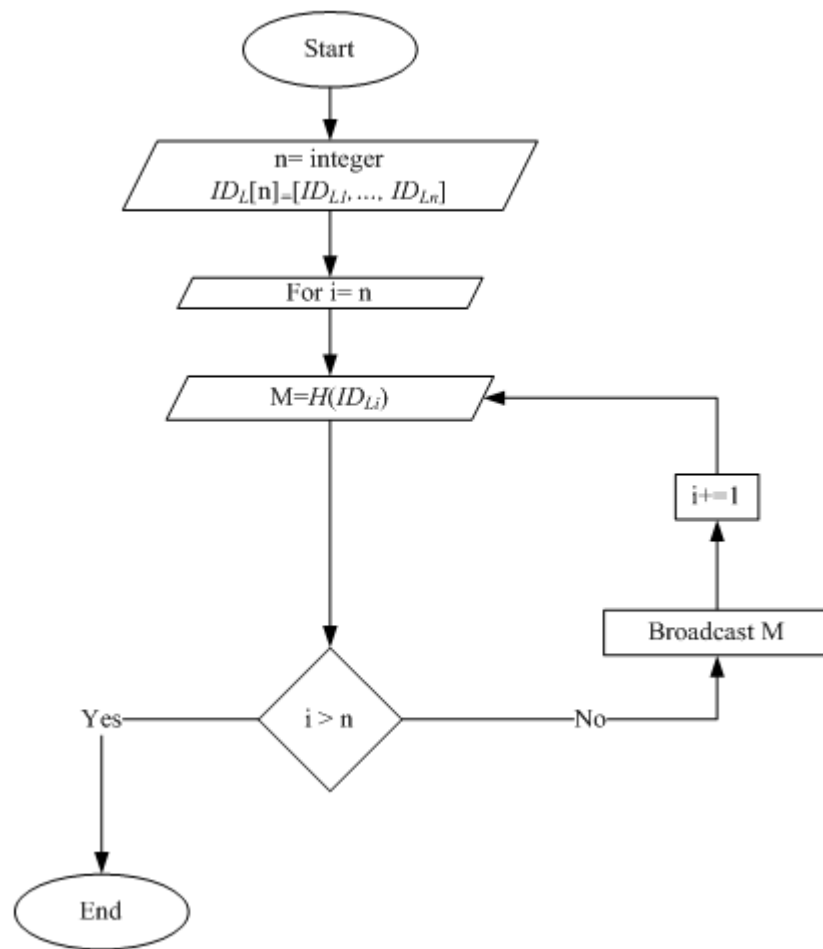
**Figure 43** Decryption of  $L_{members}$  messages by each  $CH$  algorithm

2.5 Each  $CH$  sensor sends the acknowledgement, as shown in Fig. 46, that is, the blue dotted line to its members, which is the  $ID_L$  that is encrypted by SHA-1, according to Eq. (4.4), Fig. 44 and Fig. 45.

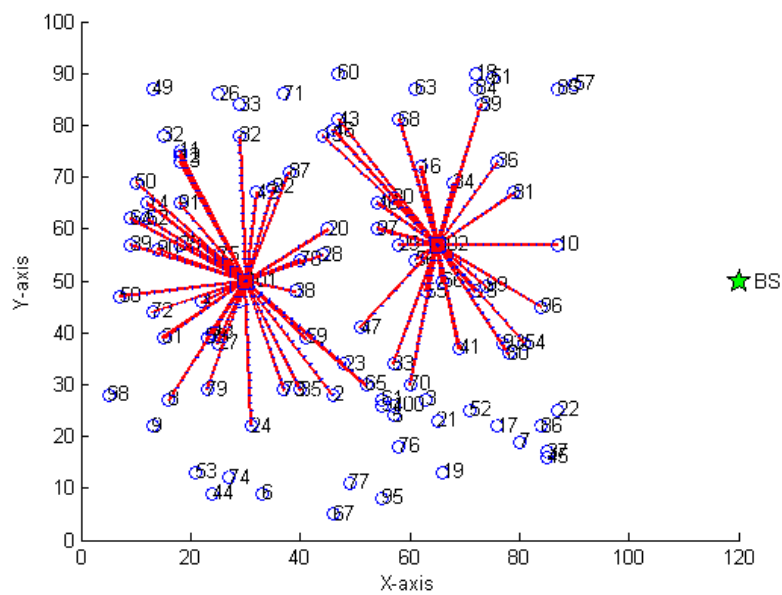
$$CH \rightarrow L: H(ID_L) \quad (4.4)$$

- 
1.  $n$ =integer.
  2.  $ID_L[n] = [ID_{L1}, \dots, ID_{Ln}]$ .
  3. For  $i=1$  to  $n$ .
    - 2.1.  $M = H(ID_{Li})$ .      Where  $M=160$  bits and  $H$  is SHA-1.
    - 2.2. Broadcast  $M$ .
  4. End.
- 

**Figure 44** The acknowledgment of  $CH$  to its member's algorithm

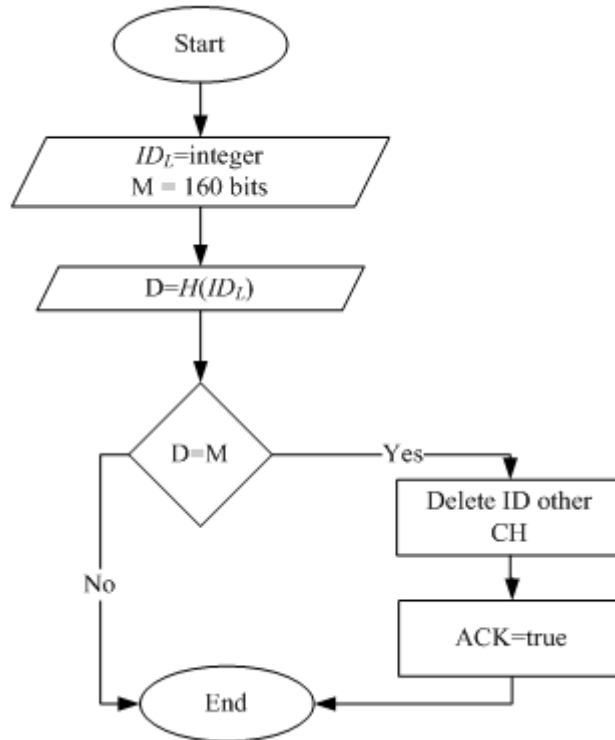


**Figure 45** The acknowledgment of CH to its member's flowchart



**Figure 46** The acknowledgment of CH to its members

2.6 Each  $L$  sensor receives 160 bits and only the  $L$  sensors can decrypt the message with their  $ID$  according to Fig. 47 and Fig. 48. Moreover, when the  $L$  sensor receives the acknowledgement, it will delete the  $ID$ s of the other  $CH$  sensors except for the second strongest to increase the memory of the  $L$  sensor.



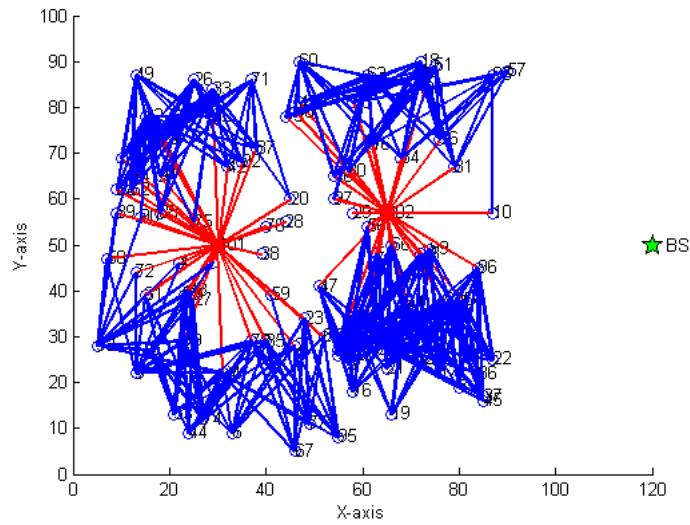
**Figure 47** The decryption acknowledgement by  $L$  sensors flowchart

- 
1.  $ID_L = \text{integer}$ .
  2.  $M = 160$ . Where its 160 bits acts the received message.
  3.  $D = H(F)$ . Where bits and  $H$  is SHA-1.
  4. If ( $D = M$ ).
    - 4.1. Delete  $ID$  other  $CH$ s.
    - 4.2. Save  $ACK$ . Where  $ACK$  is Acknowledge.
    - 4.3. End.
  5. Else
    - 5.1. Out "Discard Messages".
  6. End.
- 

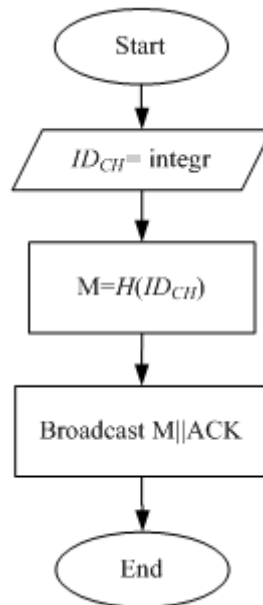
**Figure 48** The decryption acknowledgement by  $L$  sensors algorithm

2.7 Each remote  $L$  sensor which doesn't receives the  $ACK$  because it is a long distance from the selected  $CH$  sensor sends a hello message to other  $L$  sensors

according to Fig. 50 and as Fig. 51 to send its information through another connected  $L$  node as shown in Fig. 49, and is depicted with blue lines.



**Figure 49** Remote  $L$  sends Hello Message



**Figure 50** Remote  $L$  sends Hello Message flowchart

- 
1.  $ID_{CH}$  = integer.
  2.  $M = H(ID_{CH})$ . Where  $M = 160$  bits and  $H$  is SHA-1.
  3. Broadcast  $M || ACK$ . Where  $ACK$  is Acknowledge.
- 

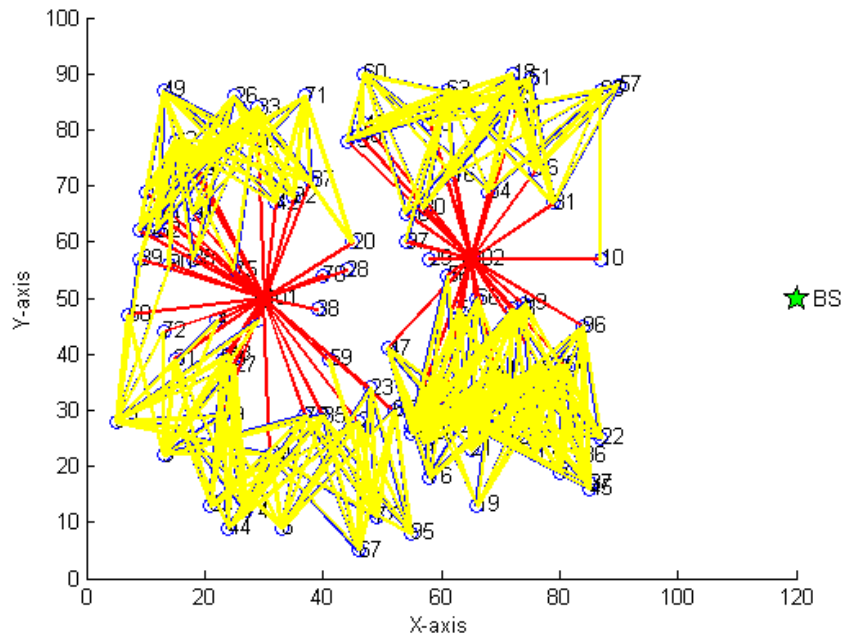
**Figure 51** Remote  $L$  sends Hello Message algorithm

2.8 The neighbor  $L$  receives the message, and before decrypting the message, it checks whether, if it has the ACK or not. If it has the ACK then it decrypts the message; otherwise, it discards the message according to algorithm in Fig. 52 and Fig. 54. Then, the neighbor  $L$  sends its  $ID_{L\_neighbor}$ , which is encrypted by  $ID_{CH}$  as shown in Fig. 53, and is depicted by using the yellow lines according to Eq. (4.5).

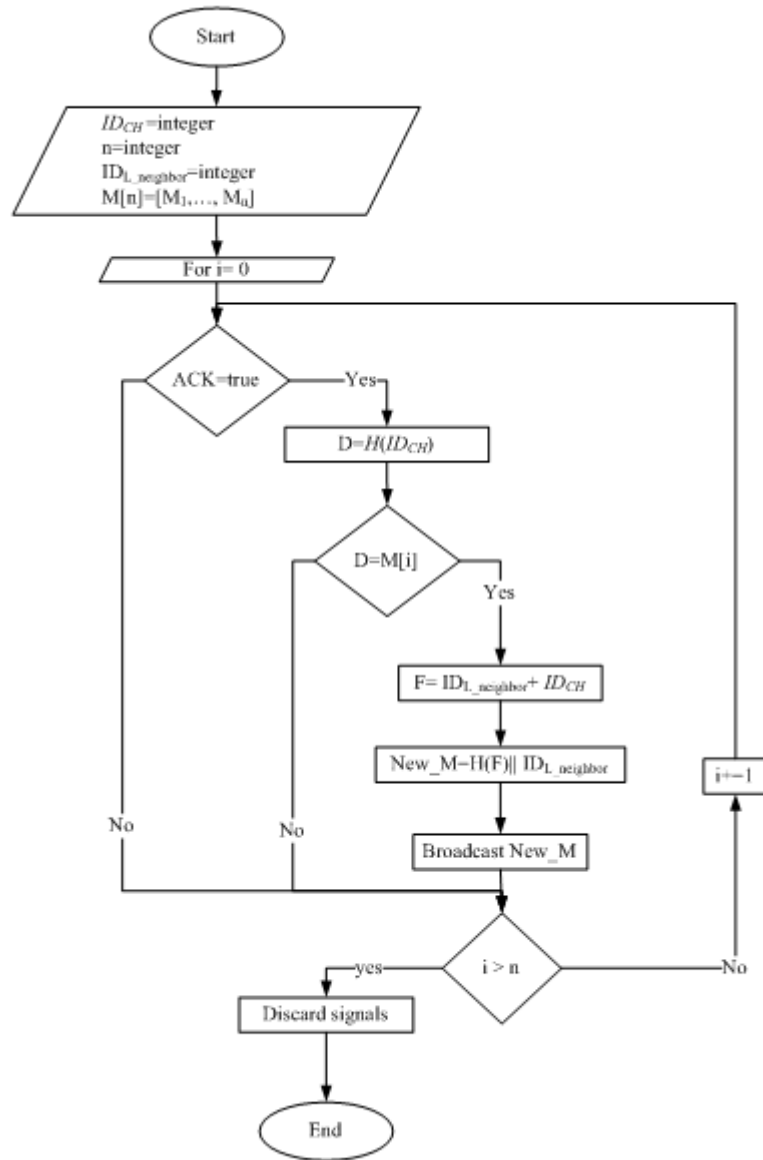
$$L_{neighbor} \rightarrow L_{far} : H(ID_{L\_neighbor} + ID_{CH}) || ID_{L\_neighbor} \quad (4.5)$$

- 
1. n=integer.
  2.  $ID_{L\_neighbor}$ =integer.
  3.  $ID_{CH}$ =integer.
  4.  $M[n]=[M_1, \dots, M_n]$ . Where each M was received messages.
  5. For i=1 to n
    - 5.1. If ACK= true.
      - 5.1.1.  $D=H(ID_{CH})$ . Where D=160 bits and H is SHA-1.
      - 5.1.2. if (D=M[i]).
        - 1.1.2.1.  $F= ID_{L\_neighbor} + ID_{CH}$ .
        - 1.1.2.2.  $New\_M=H(F) || ID_{L\_neighbor}$ .
        - 1.1.2.3. Broadcast New\_M.
      - 5.1.3. Else.
        - 5.1.3.1. Discard M.
  6. End.
- 

**Figure 52** Decryption of hello message with ACK by neighbor  $L$  algorithm



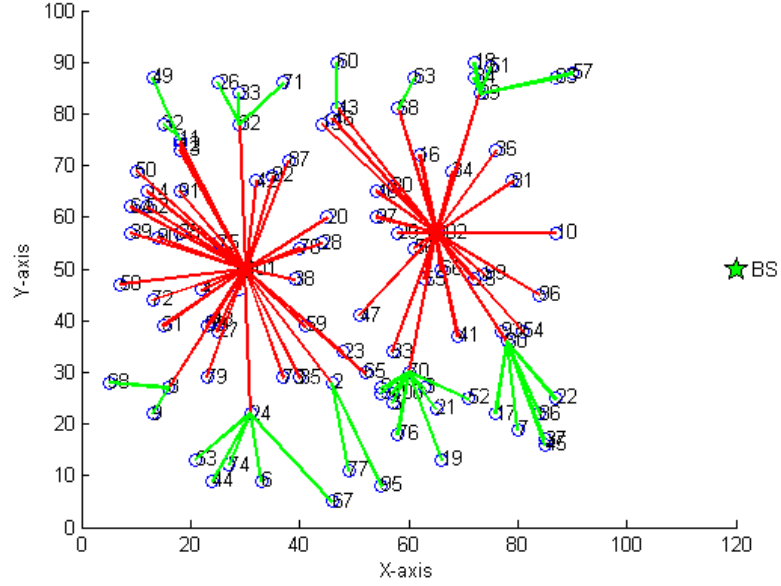
**Figure 53** The neighbor  $L$  sends its  $ID$  to remote  $L$



**Figure 54** Decryption of hello message with Ack by neighbor  $L$  flowchart

2.9 Each remote  $L$  sensor decrypts  $New\_M$  and then selects the strongest one to transport its message to its  $CH$  sensor as in the algorithm in Fig. 31 and the flowchart in Fig. 32. However, instead of using  $Share\_Key$  it uses  $ID_{L\_neighbor}$  and  $ID_{BS}$  uses  $ID_{CH}$ . Then, it sends its  $ID_{L\_remote}$  to  $L_{neighbor}$ , as shown in Fig. 53 as in the algorithm in Fig. 29 and the flowchart in 30. However, instead of using  $Share\_Key$  it uses  $ID_{L\_neighbor}$  and uses  $ID_{CH}$  instead of using  $ID_{BS}$ , as in Eq. (4.6). Finally,  $L_{far}$  sends its  $ID_{L\_far}$  to the  $CH$  sensor via  $L_{neighbor}$  as shown in Fig. 55 which is depicted by using green lines and waiting for the acknowledgement from  $CH$ .

$$L_{far} \rightarrow L_{neighbor} : H(ID_{L_{far}} + ID_{CH}) || ID_{L_{far}} \quad (4.6)$$



**Figure 55** The remote  $L$  chooses strongest signal from neighbore  $L$

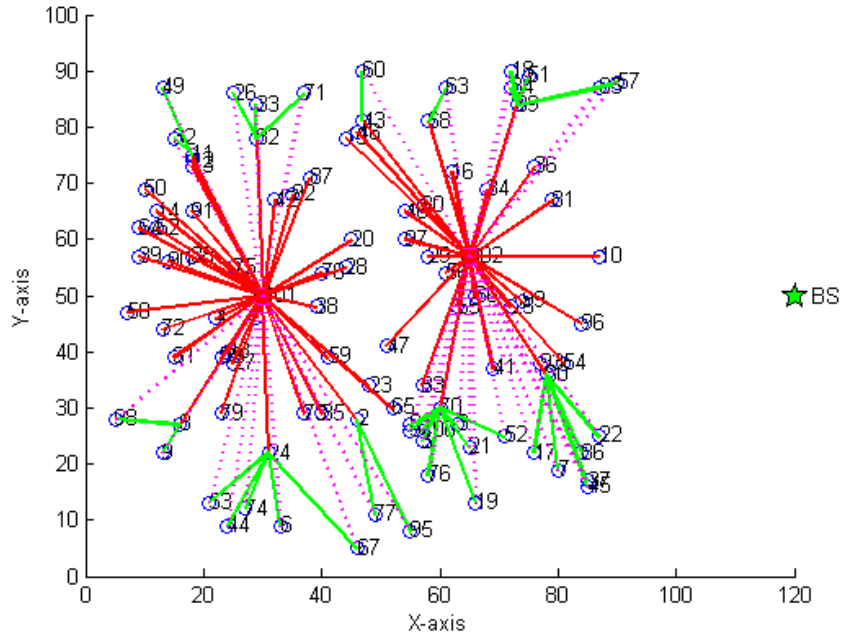
2.10 Each  $L_{neighbor}$  sensor receives and decrypts the message, and then it saves the  $ID_{L_{remote}}$ , as in the algorithm in Fig. 31 and the flowchart in Fig. 32. However, instead of using  $Share\_Key$  it uses  $ID_{L_{far}}$  and rather than using  $ID_{BS}$  it uses  $ID_{CH}$ . Then it resends the same message according Eq. (4.7).

$$L_{neighbor} \rightarrow CH : H(ID_{L_{far}} + ID_{CH}) || ID_{L_{far}} \quad (4.7)$$

2.11 Each  $CH$  sensor receives new request from  $ID_{L_{remote}}$  via  $L_{neighbor}$ , as shown in Fig. 55. The  $CH$  sensors can decrypt the message and gain the  $ID_{L_{remote}}$  as in the algorithm in Fig. 42 and the flowchart in Fig. 43 of Step 2.4. Finally, each  $CH$  sensor sends the acknowledgement which is  $ID_{L_{remote}}$  that is encrypted by SHA-1 according Eq. (4.8), as in the algorithm in Fig. 44, the flowchart in Fig. 45 of Step2 (2.5) directly to  $ID_{L_{remote}}$  as shown in Fig. 56 which is depicted by using the dotted pink lines.

$$CH \rightarrow L_{far} : H(ID_{L_{far}}) \quad (4.8)$$



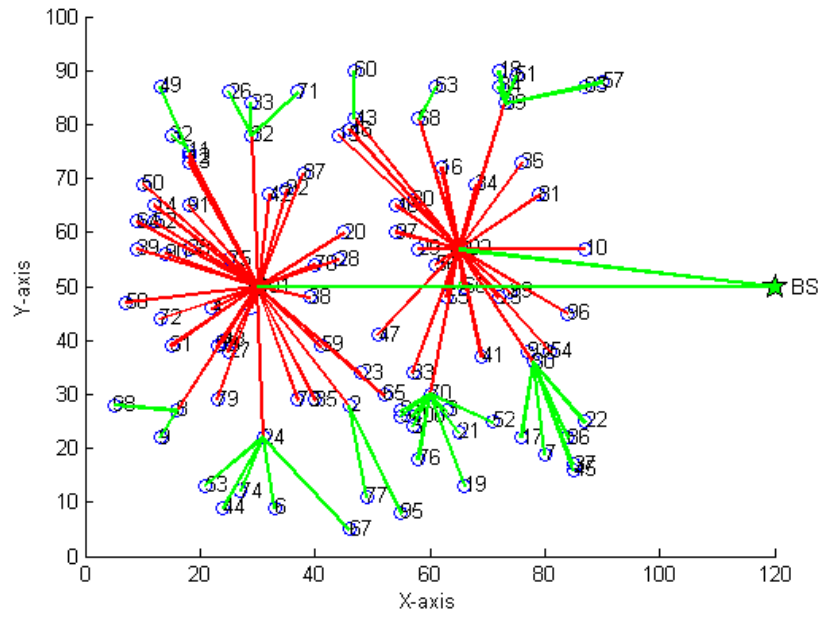


**Figure 56** The acknowledgment of  $CH$  to the remote  $L$

2.12 Each  $CH$  sensor sends a list of  $IDs$  for their members to the  $BS$ , as shown in Fig. 57 to check the  $IDs$  of the  $L$  sensors, according to Eq. (4.9).

$$CH \rightarrow BS : H(ID_{L1} + ID_{L2} + \dots + ID_{Ln}) \| ID_{L1} * ID_{L2} * \dots * ID_{Ln} \quad (4.9)$$

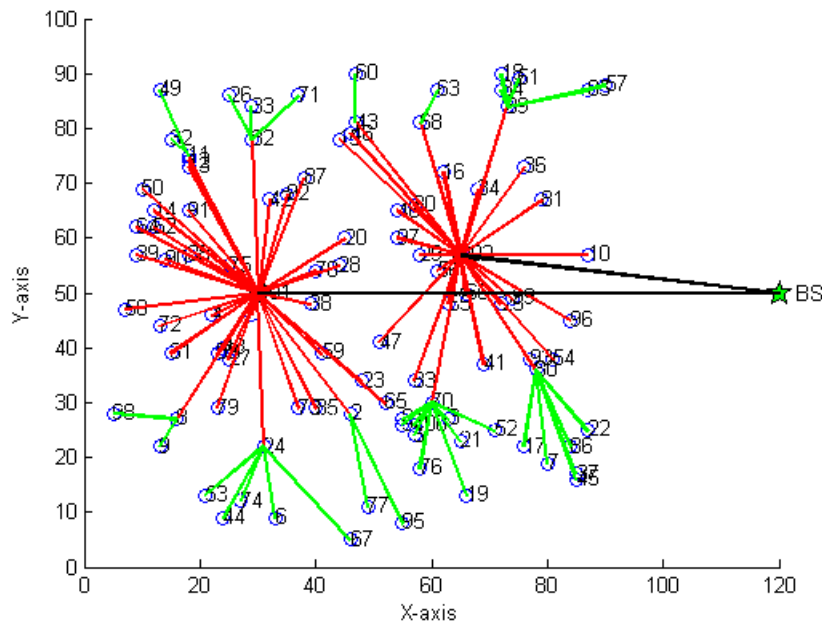
When the  $BS$  has all  $ID_L$  sensors, it checks them sequentially to recognize any illegal nodes. If there is no malicious node, the  $BS$  sends an ACK which is  $ID_{CH}$  for each  $CH$  sensor, as shown in Fig. 58, and is depicted using black lines, Eq. (4.10), as the algorithm in Fig. 44 and the flowchart in Fig. 45 of Step2 (2.5). However, instead of using  $ID_L$  it uses  $ID_{CH}$  and sends the  $ID_{L\_malicious}$  to prevent the  $CH$  sensors from exchanging the message with it.



**Figure 57** Each *CH* sends a list of members' IDs to the *BS*

Finally, each *CH* sensor receives the acknowledgement and saves the ACK as true to start with step3.

$$BS \rightarrow CH : H(ID_{CH}) \quad (4.10)$$



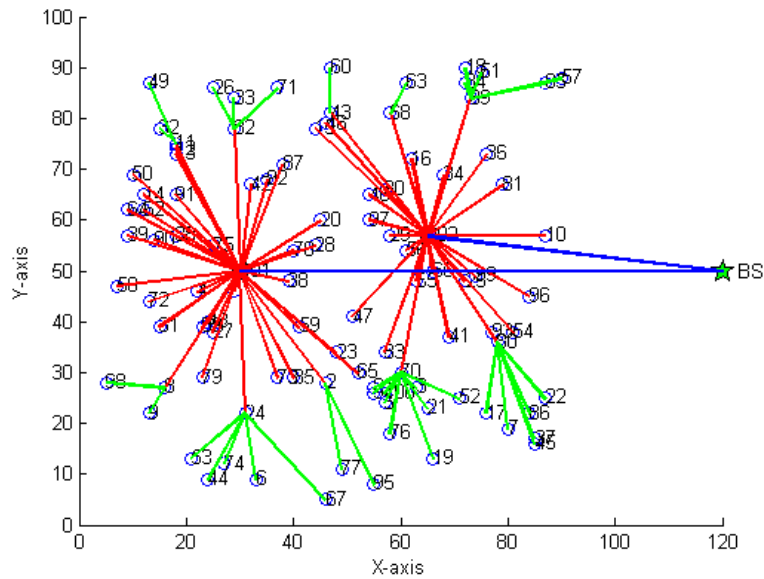
**Figure 58** The acknowledgment of *BS* to the *CH*

### Step 3. Generating and sending the *Private\_Key* to the CHs via BS

3.1 The BS generates a random integer number as *Private\_Key* for each CH sensor. Then the BS sends it to them, as shown in Fig. 59, and is depicted by using blue lines by Eq. (4.11) and the algorithm as in Fig. 60.

$$BS \rightarrow CH: MAC[E(Private\_Key_{CH} + ID_{CH}) || Private\_Key_{CH}] \quad (4.11)$$

where *E*: is the meaning of the encryption by the EDS-ECC algorithm.



**Figure 59** The BS sends the encrypt *Private\_Key* to the CHs

- 
1. EEDS-ECC algorithm steps as in Fig.21.
  2.  $n = \text{integer}$ .
  3.  $ID_{CH} = \text{integer}$ .
  4.  $ID_{CH}[n] = [ID_{CH1}, \dots, ID_{CHn}]$ .
  5. For  $i=1$  to  $n$ .
    - 5.1. EEDS-ECC algorithm step2.
    - 5.2.  $B = ID_{CHi} + Private\_Key$ .
    - 5.3.  $F = B * P$ .  $F$  is point  $B$  by ECC.
    - 5.4. EEDS-ECC algorithm step4. Where  $D = \text{convert } F$ .
    - 5.5. EEDS-ECC algorithm step7. Where  $Y = \text{convert } D$ .
    - 5.6. EEDS-ECC algorithm step8,9.
    - 5.7.  $M = W || Private\_Key$ .
    - 5.8. Broadcast  $M$ .
  6. End
- 

**Figure 60** BS sends the encrypted *Private\_Key* to the CHs algorithm

3.2 Each *CH* sensor receives a message *M*. Moreover, each *CH* sensor creates Tab. 2 through the key ring to seed key (5,7) to decrypt the generated number. The key ring size is fixed and it is 30 operations. It leads to obtaining the *Private\_Key* of each *CH* sensor. Then, each *CH* sensor deletes the key rings and retains the seed key (5,7) in their memory to reduce the memory size and prevent an adversary from getting the EC points. Finally, each *CH* sensor computes its public key, as the algorithm in Fig. 61.

---

```

1. n=integer,  $ID_{CH}$ =integer.
2.  $M[n]=[M_1, \dots, M_n]$ . Where each M was received message.
3. DEDS-ECC algorithm step 1 as in Fig. 22.
4. For i=1 to n.
    4.1.  $K$ =length of  $M[i]$ .
    4.2. For j=7 to k.
        4.2.1.  $Private\_key[j]=[M_j]$ .
    4.3. For t=1 to 6.
        4.3.1.1.  $X1[t]=M[t]$ .
    4.4. DEDS-ECC algorithm step 5,6,7,8.
    4.5.  $W=N-Private\_key$ .
        4.5.1. If ( $W=ID_{CH}$ ).
            4.3.5.1. Save Private_key.
            4.3.5.2.  $Q_{CH}=Private\_Key * P$ . Where  $Q_{CH}$  is the public key.
            4.3.5.3. End.
        4.5.2. Else.
            4.4.1. Discard  $M[i]$ .
5. End

```

---

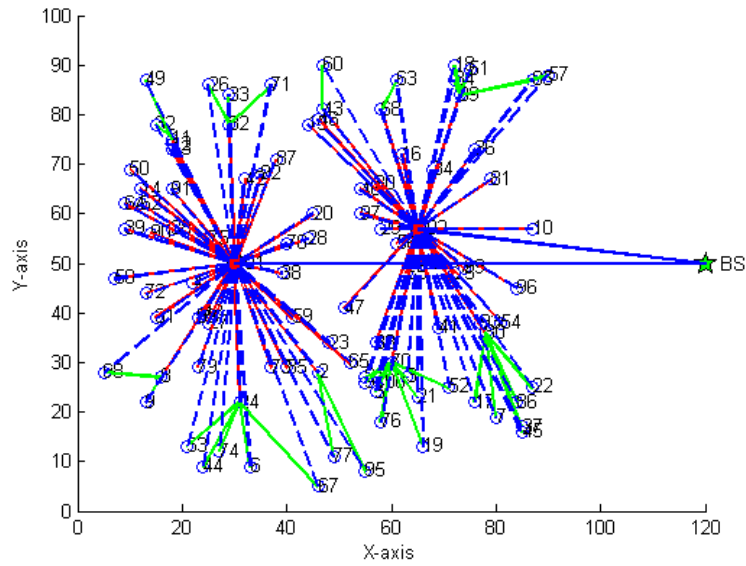
**Figure 61** The *CH*'s Extraction of the *Private\_Key* algorithm

Where *D*: is the meaning of the Decryption by EDS-ECC algorithm.

#### **Step 4. Generating and sending the *Private Key* to $L_{members}$ via *CHs***

4.1 Each *CH* sensor generates a random integer number for each *L* sensor. Additionally, to encrypt the generated number, each *CH* sensor creates Tab. 2 which acts as a key ring of the seed key (5,7) . The key ring size is fixed 30 and then sends it to them, as shown in Fig. 62, and is depicted by using dotted blue lines by Eq. (4.12) and as the algorithm in Fig. 60. However, instead of using  $ID_{CH}$ , it uses  $ID_L$ . Finally, each *CH* sensor deletes the key rings and retains the seed key (5,7) in their memory to reduce the memory size and prevent the an adversary from acquiring the EC points.

$$CH \rightarrow L: MAC[E( Private\_Key_L + ID_L )|| Private\_Key_L] \quad (4.12)$$



**Figure 62** The CHs send the encrypt *Private\_Key* to its members

4.2 Each *L* sensor receives the message *M*. To decrypt the generated number each *L* sensor creates the Tab. 2 which acts as a key ring of the seed key (5,7) . The key ring size is fixed and it is 30 operations. Then they decrypt it to get its *Private\_key*, as the algorithm in Fig. 61. However, instead of using  $ID_{CH}$ , it uses  $ID_L$ . Each *L* sensor deletes the key ring and retains the seed key (5,7) in their memory to reduce the memory size and prevent the an adversary from acquiring the EC points. Finally, each *L* sensor computes its public key.

#### 4.4.3 Update phase

The update phase is important in order to change the seed key, which is preloaded in each (*CH* and *L*) sensor. It begins before the round is finished. The base station selects another seed key and then sends it to each *CH* sensor by encrypting it with the *ID* of the *BS* and the used *Share\_Key*. Moreover, when the *CH* sensors receive the messages, they decrypt the messages and then replace the old seed key with a new seed key. Furthermore, each *CH* sensor sends the ACK to the *BS*. The *BS* generates a new *Share\_Key*. After encrypting it with the *ID* of the *BS*, each *CH* sensor saves the new decrypted *Share\_Key*. Each *CH* sensor sends the new seed keys to it members.

The update phase prevents the adversary from gaining the seed key when he captures some nodes since the seed key changes in each updated phase.

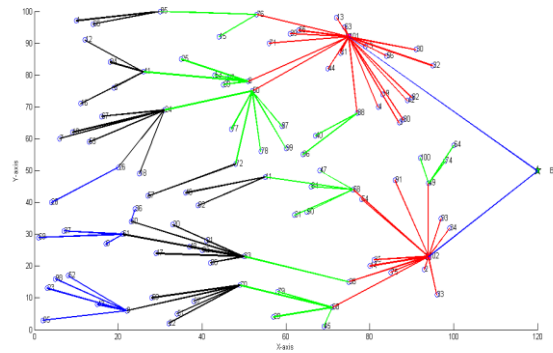
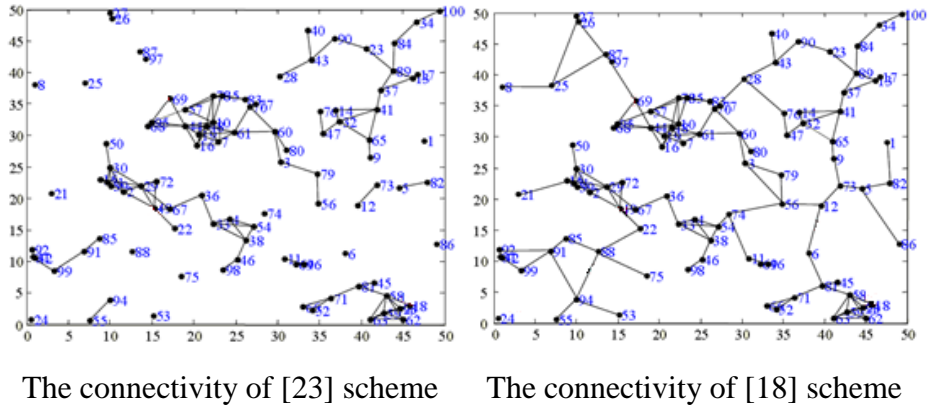
#### **4.5 Security Analysis and Results**

The proposed scheme has been simulated on a PC with an Intel(R) Core(TM) i3-2328M CPU @ 2.20GHz 2.20 GHz processor, A memory (RAM) of 4.00 GB, a 32-bit operating system *Windows 7 Ultimate* and the *MATLAB R2013a* program. In addition, the performance timings have been recorded in seconds. The security analysis of the proposed scheme encompasses the following:

##### **4.5.1 Security goals analysis:**

The security goals analysis is the more important part in our proposed scheme as given below:

1. **Authentication:** This is achieved because each node has a unique ID, as well as the use of HMAC in the distribution phase which acts as a one way function and PKC in private key generation phase.
2. **Confidentiality:** The proposed scheme encrypts every message and makes it secrets by mean of the proposed algorithms.
3. **Integrity:** The proposed scheme protects the received messages from alteration and modification it by using HMAC in the proposed algorithms.
4. **Scalability:** The proposed scheme is sufficiently flexible to increase in size after deployment without the influence of network performance.
5. Due to the repeated private keys, the proposed scheme is resistant against node capture (higher resilience means a lower number of compromised links). When an attacker captures some nodes, he cannot hack the network until every node is captured.
6. **Connectivity:** in the proposed scheme every node communicates to create the network, as shown in Fig. 63 Moreover, hops numbers in the proposed scheme are lower than the compared scheme and the connectivity rate is 100%. Finally, the proposed scheme scatter 100 nodes in a  $100 * 100 \text{ m}^2$  area while in [18-23], it scatter 100 nodes in a  $50 * 50 \text{ m}^2$  area.



**Figure 63** The connectivity comparison

Finally, Tab. 9 depicts the comparison between the proposed scheme with other schemes in terms of security goals.

**Table 9** Comparison in Terms of Security Goals

Scheme	The Proposed Scheme	Scheme [56]	Scheme [24]
<b>Authentication</b>	One-way	One-way	One-way
<b>Confidentiality</b>	Maintain	Not maintain	Not maintain
<b>Integrity</b>	Maintain	Not maintain	Not maintain
<b>Cryptographic technique</b>	Symmetric based on HMAC and Asymmetric based on ECC	Self-certified key (SCK)	Hash and XOR
<b>Scalability</b>	Yes	Yes	Yes

#### 4.5.2 Attacks analysis:

The proposed scheme is active against the following attacks:

1. Eavesdrops attack acquires the signal. It does not conclude what is inside the message because the message content is a stream of bits. Moreover, the message was encrypted since it does not know the encryption algorithm to extract the original message.
2. Sybil attack sends a message to legal nodes within the network with a fake *ID*. Each *CH* sensor and *L* sensor can detect this attack because it has the *IDs* of its member and discards the malicious node *ID* through *ID* verification.
3. The proposed scheme discards the sinkhole attack because only the *CH* sensors can communicate with the *BS*. Furthermore, it uses Share Key to exchange the messages, where the attacker cannot know the Share Key.
4. Hello flood attack cannot affect the proposed scheme because it uses a hello message with  $H(ID_{CH})$ . Furthermore, the *ID* of *CH* sensors are preloaded into each *L* sensor. However, the attacker message can be recognized from the nodes within the network. Therefore, it will be discarded.
5. Wormhole attacks want to occur between two legal nodes and exchange fake messages with each legal node. Each *CH* and *L* can detect this attack because it uses HMAC to verify the message and discard any malicious node messages.
6. When the enemy compromises a node, for example an *L* sensor, and makes copies of this node, it deploys these nodes in the network. Initially, it communicates with the *CH* sensors. However, afterwards each *CH* sensor sends a list of members' *IDs* to the *BS* and the *BS* knows that a node was compromised by the node's *ID* having been duplicated. The *BS* sends messages to the *CH* sensors in order to avoid dealing with this node.

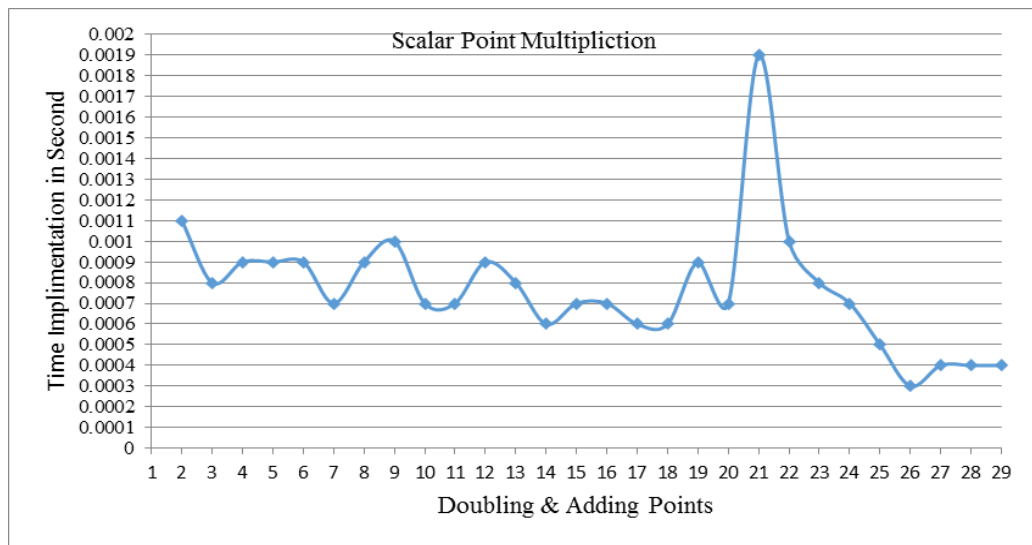
In addition, we compare the proposed scheme with some existing schemes in term of attacks as shown in Tab (10).



**Table 10** Comparison in Terms of Attack Security

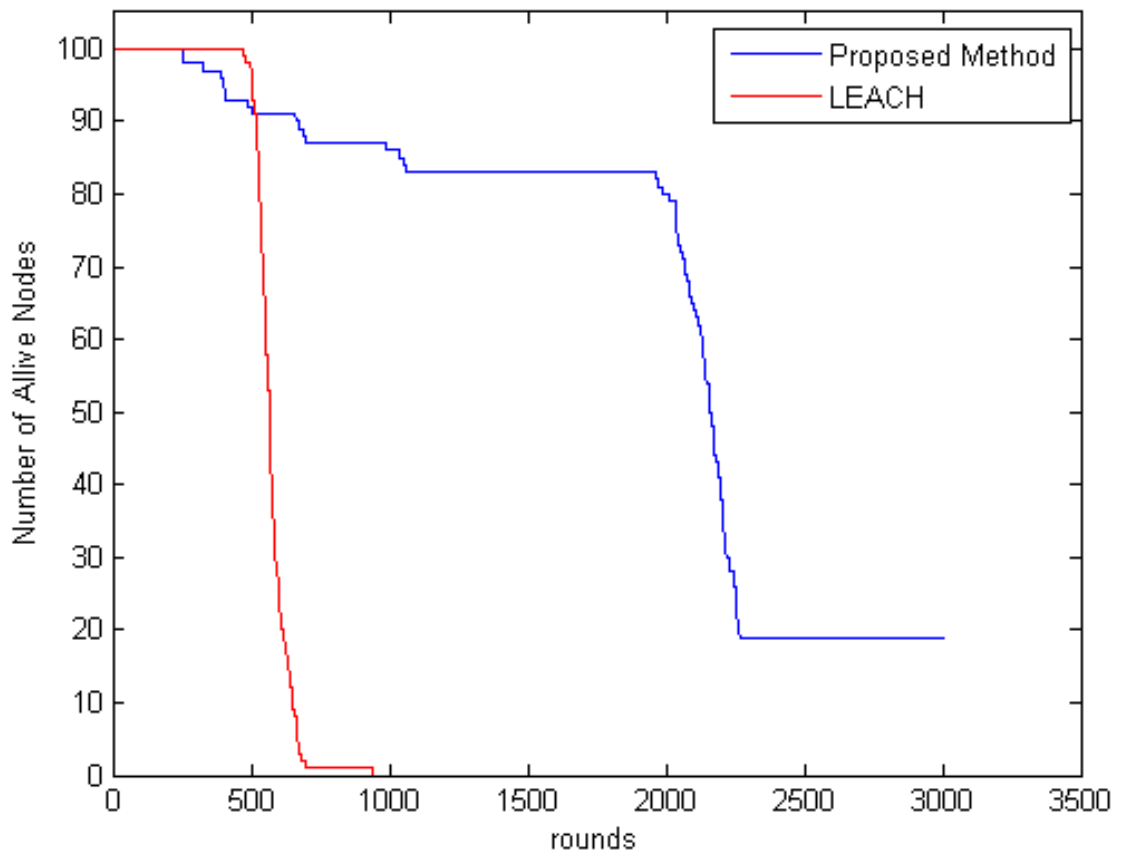
Attack \ Scheme	Scheme [57]	Scheme [58]	Proposed scheme
Eavesdropping	No	No	Yes
Hello flood	No	Yes	Yes
Worm-hole	Yes	Yes	Yes
Sink hole	Yes	Yes	Yes
Sybil	Yes	No	Yes
Clone	No	No	Yes

Moreover, there is scalar point multiplication in terms of seconds, as shown in Fig. 64, which shows the key ring for the seed key (5,7) which is needed to decrypt the receiving private key from *BS* to each *CH* sensor. Additionally, it utilizes *CH* sensors to encrypt the sending private key to their members. Each *L* sensor using the key ring to decrypt receives a private key from the *CH* sensors.



**Figure 64** The key ring (30 operations) for the seed key in term of second

Finally, Fig. 65 shows the life time of sensors. It can be seen that the proposed scheme performance better than the LEACH scheme [22] in terms of number of dead sensors. In the proposed scheme sensors are dead after 2250 round. However, in the LEACH scheme, the sensors are dead after 750 rounds.



**Figure 65** The comparison proposed scheme with LEACH

## CHAPTER 5

### CONCLUSION

Key management is a crucial part and a fundamental issue in WSNs. In addition, it is essential to establish secure communication among the network elements by using cryptographic techniques. Because of sensor resource constraints, it is very difficult to address all the security requirements in WSNs and enemy threats. The proposed scheme uses the SHA-1 in HMAC to form the cluster. It encrypts the private key via [52], which decreases the number of each point to 6 bits rather than 8 bits. The proposed scheme should be used to provide acceptable energy consumption, connectivity, security goals, and resistance upon threads when compared with another scheme. Moreover, it prevents an adversary from acquiring these keys in cases of node capture. These keys will be changed in the updated phase. Furthermore, the *BS* generates the private key for *CH* sensors and the *CH* sensors generate the private key for *L* sensors. However, in [59], the *BS* is responsible for generating the private for each node. Finally, the proposed scheme is more scalable and resilient.

## REFERENCES

1. **Sanjit K. D., Subasish M., Prasant K. P., (2010)**, "A Survey on Application of Wireless Sensor Network Using Cloud Computing," *International Journal of Computer Science & Engineering Technologies (E-ISSN: 2044-6004)*, vol. 1, pp. 50-55.
2. **Liljana G., Srdkjan K., Veljko M., Ivan S., Roman T., (2011)**, "Application and Multidisciplinary Aspects of Wireless Sensor Networks: Concepts, Integration, and Case Studies", Springer, London, pp. 3,25,89-90.
3. **Waltenegus D., Cristian P., (2010)**, "Fundamentals of Wireless Sensor Networks: Theory and Practice", John Wiley & Sons, New Jersey, pp. 8,14.
4. **Kazem S., Daniel M., Taieb Z., (2007)**, "Wireless Sensor Networks: Technology, Protocols, and Applications", John Wiley & Sons, New Jersey, pp. 37-38.
5. **Ravi K. K., (2014)**, "Key Management Technique for WSNs," in Region 10 Symposium, IEEE, pp. 540-545.
6. **Khadija R., Nujhat N., Al-Sakib K. P., (2010)**, "An Enhanced Tree-Based Key Management Scheme for Secure Communication in Wireless Sensor Network," in High Performance Computing and Communications (HPCC), IEEE International Conference on, pp. 671-676.
7. **Xuefei S., Achim L., Zisis G., Berthold S., Moatasem E., (2014)**, "Effect of Biometric Characteristics on the Change of Biomechanical Properties of the Human Cornea due to Cataract Surgery," *BioMed Research International*, pp. 1-5.

8. **Zhang Y., Ji P., (2014),** "*An Efficient and Hybrid Key Management for Heterogeneous Wireless Sensor Networks,*" in Control and Decision Conference (CCDC), The 26th Chinese, pp. 1881-1885.
9. **Ayman T., Ayman K., Ali C., (2014),** "*Authentication Schemes for Wireless Sensor Networks,*" in Mediterranean Electrotechnical Conference (MELECON), IEEE, pp. 367-372.
10. **Abdulaziz A. A., Imad F. A., Mohammad A. A., (2013),** "*Cryptographic Hash Function: A High Level View,*" in Informatics and Creative Multimedia (ICICM), International Conference on, pp. 128-134.
11. **Florian L., Gilles D., Michael K., (2012),** "*Encoding Hash Functions as a SAT Problem,*" in Tools with Artificial Intelligence (ICTAI), IEEE 24th International Conference on, pp. 916-921.
12. **Iftekhhar S., Pardeep K., HoonJae L., (2010),** "*An Efficient Key Pre-Distribution Scheme for Wireless Sensor Network Using Public Key Cryptography,*" in Networked Computing and Advanced Information Management (NCM), Sixth International Conference on, pp. 402-407.
13. **Christoph K., Markus S., Claudia E., (2008),** "*On Handling Insider Attacks in Wireless Sensor Networks,*" in Information Security Technical Report, vol. 13, pp. 165-172.
14. **Hai Y., Zhijie J. S., Yunsi F., (2009),** "*Efficient Implementation of Elliptic Curve Cryptography on DSP for Underwater Sensor Networks,*" in 7th Workshop on Optimizations for DSP and Embedded Systems (ODES-7), pp. 7-15.
15. **Taogai Z., Hongshan Q., (2010),** "*A Lightweight Key Management Scheme for Wireless Sensor Networks,*" in Education Technology and Computer Science (ETCS), Second International Workshop on, pp. 272-275.
16. **Song j., (2012),** "*A Lightweight Key Establishment in Wireless Sensor Network Based on Elliptic Curve Cryptography,*" in Intelligent Control, Automatic Detection and High-End Equipment (ICADE), IEEE International Conference on, pp. 138-141.

17. **Yiying Z., Xiangzhen L., Jianming L., Jucheng Z., Baojiang C., (2012),** "A Secure Hierarchical Key Management Scheme in Wireless Sensor Network," International Journal of Distributed Sensor Networks, pp. 1-8.
18. **Kishore R., Radha S., Ramasamy P., (2011),** "A Secure Key Predistribution Scheme for WSN Using Elliptic Curve Cryptography," ETRI Journal, vol. 33, pp. 791-801.
19. **Yuan Z., Yongluo S., Sang K. L., (2010),** "A Cluster-Based Group Key Management Scheme for Wireless Sensor Networks," in Web Conference (APWEB), 12th International Asia-Pacific, pp. 386-388.
20. **Qing Y., Qiaoliang L., Sujun L., (2008),** "An Efficient Key Management Scheme for Heterogeneous Sensor Networks," in Wireless Communications, Networking and Mobile Computing, WiCOM '08, 4th International Conference on, pp. 1-4.
21. **Rahman M., El-khatib K., (2010),** "Private Key Agreement and Secure Communication for Heterogeneous Sensor Networks," Journal of Parallel and Distributed Computing, vol. 70, pp. 858-870.
22. **Wendi R. H., Anantha C., Hari B., (2000),** "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in System Sciences. Proceedings of the 33rd Annual Hawaii International Conference on, vol. 2, pp. 1-10.
23. **Rolf B., (1985),** "An Optimal Class of Symmetric Key Generation Systems," in Advances in Cryptology, pp. 335-338.
24. **Canming J., Bao L., Haixia X., (2007),** "An Efficient Scheme for User Authentication in Wireless Sensor Networks," in Advanced Information Networking and Applications Workshops, AINAW'07, 21st International Conference on, pp. 438-442.
25. **Amounas F., El Kinani E., (2012),** "ECC Encryption and Decryption with a Data Sequence," Applied Mathematical Sciences, vol. 6, pp. 5039-5047.
26. **Rehana Y., (2012),** Ph.D. Thesis, "An Efficient Authentication Framework for Wireless Sensor Networks," Dept. School of Computer Science, School

of Computer Science College of Engineering and Physical Sciences,  
University of Birmingham, United Kingdom, pp. 4,16-17,70,112.

27. **Lalitha T., Devi A. J., (2014),** "*Security in Wireless Sensor Networks: Key Management Module in EECBKM,*" in Computing and Communication Technologies (WCCCT), World Congress on, pp. 306-308.
28. **Elaine B., William B., William B., William P., Miles S., (2012),** "*Recommendation for Key Management-part 1: General (Revised),*" in NIST Special Publication, Department of Commerce, United States of America, pp. 95,103.
29. **Heena S., Awan D., (2013),** "*An Enhanced and Efficient Mechanism to Detect Sybil Attack in Wireless Sensor Networks,*" in International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 2, pp. 792-796.
30. **Gaurav G., Nikita V. M., (2014),** "*Securing Multipath Routing Protocol Using Authentication Approach for Wireless Sensor Network,*" in Communication Systems and Network Technologies (CSNT), Fourth International Conference on, pp. 729-733.
31. **Vamsi P. R., Kant K., (2014),** "*A Lightweight Sybil Attack Detection Framework for Wireless Sensor Networks,*" in Contemporary Computing (IC3), Seventh International Conference on, pp. 387-393.
32. **Kulkarni G., Shelk R., Gaikwad K., Solanke V., Gujar S., Khatawkar P., (2013),** "*Wireless Sensor Network Security Threats,*" in Communication and Computing (ARTCom), Fifth International Conference on Advances in Recent Technologies, pp. 131-135.
33. **Manali D. S., Shrenik N. G., Narendra M. S., (2014),** "*Lightweight Authentication Protocol Used in Wireless Sensor Network,*" in Circuits, Systems, Communication and Information Technology Applications (CSCITA), International Conference on, pp. 138-143.
34. **Behrouz A. F., (2013),** "*Data Communications and Networking*", McGraw-Hill, New York, pp. 933-955.

35. **Ibrahim A., (2011)**, "A Key Distribution and Management Scheme for Hierarchical Wireless Sensor Network," International Journal of Multimedia & Ubiquitous Engineering, vol. 6, pp. 1-12.
36. **Selvam R., Senthilkumar A., (2014)**, "Cryptography Based Secure Multipath Routing Protocols in Wireless Sensor Network: A Survey," in Electronics and Communication Systems (ICECS), International Conference on, pp. 1-5.
37. **Kanika G., Alvin L., (2015)**, "A Survey of Broadcast Authentication Schemes for Wireless Networks," Ad Hoc Networks, vol. 24, pp. 288-316.
38. **Abduvaliyev A., Sungyoung L., Young-Koo L., (2009)**, "Modified SHA-1 Hash Function (mSHA-1)," in ITC-CSCC: International Technical Conference on Circuits Systems, Computers and Communications, pp. 1320-1323.
39. **Scott V., Alfred M., (2004)**, "Guide to Elliptic Curve Cryptography", Springer, New York, pp. 1,11-12,18,26.
40. **Florian H., Andreas S., Sandra S., Manfred L., (2012)**, "The Magic of Elliptic Curves and Public Key Cryptography", Springer-Verlag, United Kingdom, p. 1.
41. **Jaydip S., (2012)**, "Cryptography and Security in Computing ", InTech, Croatia, p. 91.
42. **Xuanxia Y., Xiaoguang H., Xiaojiang D., (2014)**, "A Light-weight Certificate-Less Public Key Cryptography Scheme Based on ECC," in Computer Communication and Networks (ICCCN), 23rd International Conference on, pp. 1-8.
43. **Mohsen B., Sharifah Y., Ramlan M., (2013)**, "Comparison of ECC and RSA Algorithm in Resource Constrained Devices," in IT Convergence and Security (ICITCS), International Conference on, pp. 1-3.
44. **Azzedine B., (2008)**, "Algorithms and Protocols for Wireless Sensor Networks", John Wiley & Sons, New Jersey, vol. 62, p. 497.



45. **Arindam S., Jyotsna K. M., (2012),** "*Secured Wireless Communication Using Fuzzy Logic Based High Speed Public-Key Cryptography (FLHSPKC)*," in International Journal of Advanced Computer Science and Applications (IJACSA), vol. 3, pp. 137-145.
  
46. **Garima V., Amandeep K., (2014),** "*A Review on Distributed System Security using Elliptic Curve Cryptography*," in International Journal of Scientific and Research Publications, vol. 4, pp. 1-6.
  
47. **Pradip K. S., Chhotray R. K., Gunamani J., Sabyasachi P., (2013),** "*An Implementation Of Elliptic Curve Cryptography*," in International Journal of Engineering Research and Technology (IJERT), vol. 2, pp. 1-8.
  
48. **Apostolos P. F., John Z., Odysseas K., (2014),** "*Designing and Evaluating High Speed Elliptic Curve Point Multipliers*," in Digital System Design (DSD), 17th Euromicro Conference on, pp. 169-174.
  
49. **Shweta L., Monika S, (2013),** "*An Efficient Elliptic Curve Digital Signature Algorithm (ECDSA)*," in Machine Intelligence and Research Advancement (ICMIRA), International Conference on, pp. 179-183.
  
50. **Lawrence. C. W., (2008),** "*Elliptic Curves: Number Theory and Cryptography*", Chapman & Hall/CRC Press, United States of America, p. 143.
  
51. **Jaydeep B., Bintu K., (2014),** "*Improvement of Deterministic Key Management Scheme for Securing Cluster-Based Sensor Networks*," in Networks & Soft Computing (ICNSC), First International Conference on, pp. 55-59.
  
52. **Aso. A. M., Kameran A. A., Ahmed C. S., Yuriy A., (2014),** "*The Enhanced Data Sequence Method for ECC Cryptosystem*," Applied Mathematical Sciences, vol. 8, pp. 5553-5564.
  
53. **Neal K., (1987),** "*Elliptic Curve Cryptosystems*," Mathematics of Computation, vol. 48, pp. 203-209.
  
54. **Jia X., Wang C., (2005),** "*The Application of Elliptic Curve Cryptosystem in Wireless Communication*," in Microwave, Antenna, Propagation and EMC

Technologies for Wireless Communications, IEEE International Symposium on, Vol. 2 , pp. 1602-1605.

55. **Kenneth H. R., (2011),** "*Discrete Mathematics and Its Applications*", McGraw-Hill, New York, pp. 156-164.
  
56. **Huei-Ru T., Rong-Hong J., Wu Y., (2007),** "*An Improved Dynamic User Authentication Scheme for Wireless Sensor Networks*," in Global Telecommunications Conference, GLOBECOM'07 IEEE, pp. 986-990.
  
57. **Sajid H., Firdous K., Ashraf M., (2007),** "*An Efficient Key Distribution Scheme for Heterogeneous Sensor Networks*," in Proceedings of the International Conference on Wireless Communications and Mobile Computing, pp. 388-392.
  
58. **Boushra M., Hatem B., Abdelmadjid B., (2008),** "*TLA: A Tow Level Architecture for Key Management in Wireless Sensor Networks*," in Sensor Technologies and Applications, SENSORCOMM'08 Second International Conference on, pp. 639-644.
  
59. **Abdullah A., Rumana A., (2012),** "*Secure Sensor Node Authentication in Wireless Sensor Networks*," International Journal of Computer Applications, vol. 46, pp. 10-17.

## APPENDICES A

### CURRICULUM VITAE

#### PERSONAL INFORMATION

**Surname, Name:** AL-SALIHI, Aso Ahmed Majeed

**Date and Place of Birth:** 1 January 1982, Kirkuk

**Marital Status:** Married

**Phone:** +90 5535 47 34 13

**Email:** asoalsalihi@gmail.com



#### EDUCATION

Degree	Institution	Year of Graduation
M.Sc.	Çankaya University, Computer Engineering	2015
B.Sc.	College of Tecnology/Kirkuk, Software Engineering Techniques	2004
High School	Al-Thawra for Boys	1999

#### FOREIN LANGUAGES

Advanced Arabic, Advanced English, Beginner Turkish.

#### PUBLICATIONS

1. Aso. A. M., *et al.*, "The Enhanced Data Sequence Method for ECC Cryptosystem," Applied Mathematical Sciences, vol. 8, pp. 5553-5564, 2014.

#### HOBBIES

Football, Travel, Books, Swimming.