Wireless Networks
https://doi.org/10.1007/s11276-019-01980-1

DSKMS: a dynamic smart key management system based on fuzzy logic in wireless sensor networks

Mohammad Sadegh Yousefpoor[1] · Hamid Barati[1]

Abstract

Wireless sensor networks include a large number of sensor nodes which are developed in an environment. These networks have military, commercial, industrial, and medical applications. Providing these networks with security is of absolute importance. One strategy for maintaining the security of these networks is key management. In the presented paper, we propose a smart dynamic key management system for wireless sensor networks which utilizes fuzzy logic for path key generation and for adding new nodes to the network. Our proposed key management system falls in the category of methods which are based on hierarchical networks and it utilizes both pre-distribution and post-deployment mechanisms of key distribution among the sensor nodes. Utilization of fuzzy logic in the proposed key management system results in increased accuracy in decision making and contributes to its smartification. As a result, energy consumption in the network is reduced and the network lifetime is enhanced. Simulation results show that our system, compared to other key management systems, is more efficient in terms of communication overload, required memory space, and energy consumption. Furthermore, our system demonstrated proper resilience and resistance to cryptanalysis attacks.

Keywords Wireless sensor networks (WSNs) · Dynamic key management · Security · Rekeying · Fuzzy logic

1 Introduction

Wireless sensor networks consist of a set of extremely light sensor nodes with limited energy resources. In these networks, a small battery is used to provide the network nodes with energy; therefore, the sensor nodes have limited energy, storage capacity and processing power [1]. The sensor nodes transfer the collected data to the base station through single-hop or multi-hop mechanisms. Wireless sensor networks have applications in the Internet of Things (IoT) [2], and also in military, commercial, industrial, and agricultural fields as well as in the supervision of the nuclear radiations and the spread of chemical warfare and in the supervision of the ecosystem, etc [3].

Wireless sensor networks, based on their network models, are either categorized as hierarchical or peer-to-peer. The hierarchical (cluster-based) model usually has a layered architecture which consists of several static or dynamic clusters [4]. This model facilitates the network management, increases the scalability of the network, and contributes to a reduction in the load of the message transmission [5]. The peer-to-peer (distributed) model is also used in many sensitive applied programs in wireless sensor networks in which each sensor node directly communicates with its neighbour node [6]. In this model, the sensor nodes are easily developed in a self-managing distributed system, but network management is difficult and the scalability of the network will not materialize [7].

The network security is a set of general policies, mechanisms and services which protects a network against attacks and unauthorized access [8]. Security in wireless sensor networks requires instruments and techniques [9]. The security process in the networks is very useful and in their absence the communication system will face problems, nevertheless these processes cause computing overload, communication overload, and an increase in the

✉ Hamid Barati
hbarati@iaud.ac.ir

Mohammad Sadegh Yousefpoor
ms.yousefpoor@iaud.ac.ir

[1] Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran

Published online: 20 March 2019

consumption of both energy and memory space [10]. The design of security systems in wireless sensor networks has always involved challenges that have to be taken into consideration. Therefore, utilizing security mechanisms that are capable of optimally using the limited capacity of the nodes and improve the security and lifetime of the network, is absolutely vital. One strategy for maintaining the security of these networks is key management [11].

Key management is defined as a set of processes and mechanisms which support key formation and allow keying among valid nodes based on a security policy and guarantees security in the services and applications of the wireless sensor networks [12]. The purpose of key management in wireless sensor networks is to address issues like the generation, distribution and support of encrypted keys [13]. In the process of key generation, encryption algorithms are used for keying among the nodes in the network. For the distribution of the keys among the sensor nodes of the network, two mechanisms could be utilized: pre-distribution and post-deployment [14]. In the key pre-distribution mechanism, the code keys are loaded into the node memories before the nodes are developed in the network [15]. But in the post-deployment keying mechanism, the process of key generation among the sensor nodes takes place after the deployment of the network and in a dynamic manner. In a key management system, the purpose of rekeying is supporting the encrypted keys [16].

Based on their ability to rekey the encrypted keys, the key management schemes are divided into static and dynamic schemes. In static key management, the rules of the keys are agreed before their distribution and the keys are constant during the lifetime of the network. Therefore, the probability of attacks to the networks will be considerably higher [17]. In dynamic key management, the encrypted keys will be rekeyed during the lifetime of the network, in order to remove the keys of cryptanalyzed nodes during the rekeying process. For this reason, dynamic key management techniques are considered by the researchers as suitable key management schemes for wireless sensor networks. Dynamic key management schemes will considerably increase the network lifetime as well as its resilience and resistance to attacks [18].

In the presented paper, we will present a dynamic smart key management system (DSKMS). This system falls in the category of methods which are based on hierarchical networks. The objectives of designing this system are: enhancing the security of the network through deleting cryptanalyzed nodes, increasing the resilience and resistance to cryptanalysis attacks and reducing the memory space used, the communication overload and the energy consumed in the network. In the design of this system, we used fuzzy logic in the process of path key generation and the process of adding new nodes to the network. This method improved the efficiency of our system and since it reduces energy consumption, it also elongates the network lifetime. The simulation results show that DSKMS is more efficient than the three key management systems of HKMS [19], LEAP+ [20], and EAHKM [21].

The rest of the paper is organized in the following manner: In Sect. 2, some of the works already done in the field of key management will be reviewed. In Sect. 3, we will state the basic concepts of the design of our key management system and will briefly explain fuzzy logic and encryption methods. In Sect. 4, we will explain the network model of our system and some of its features. In Sect. 5, we thoroughly expound our system and in detail will explain each process involved in it. In Sect. 6, we will evaluate the security of the system and will assess its effectiveness with security scales such as forward and backward secrecy, resilience and resistance to attacks. In part 7, the results obtained from simulation of DSKMS will be compared to that of HKMS [19], LEAP+ [20], and EAHKM [21]. Finally in part 8, we will present the conclusion of our paper.

2 Related works

In this part, based on the hierarchical and peer-to-peer network models introduced in Sect. 1, we will review a number of key management systems.

Ahlawat et al. [22] presented a secure key management scheme in wireless sensor networks. The purpose of this method is to reduce the impact of node capture on the entire network by using an efficient adversarial model. In this scheme, the network model is peer to peer. In the proposed scheme by Priyanka Ahlawat et al., symmetric key cryptography method is used for key generation. In this method, random pairwise keys are employed for a secure connection between nodes and key pre-distribution method is used for distributing keys among sensor nodes. If two neighbouring nodes have at least one common key in their related key sources, they can secure a connection between each other. Rekeying process is done periodically or at the time of node compromise in the network.

Aissani et al. [23] suggested a dynamic key management method for wireless sensor networks. Authors in this scheme, believe that many existing solutions focus on optimizing key number, the frequency of rekeying process and key cryptography processes. In fact, a key management method is implemented like an independent service, so it suffers from overload. Therefore, Sofiane Aissani et al. reached to this conclusion to propose μKMS in order to implement the dissimulation scheme and embed rekeying process messages on the unused coding space of exchanged

ZigBee packets. For this reason, μKMS is an overload-free key management scheme.

Seo et al. [24] proposed a key management system called Certificateless effective key management protocol (CL-EKM). In this system, the model of the wireless sensor network is dynamic, heterogeneous, and hierarchical. Moreover, in this system rekeying process is introduced in order to protect the network against cryptanalysis attacks. CL-EKM supports both node movement and adding new nodes to the network. Furthermore, this system guarantees confidentiality, integrity and forward and backward secrecy, and is resistant to impersonation attacks, node cloning and node capture attacks.

Blom [25] introduced another key management method. In this method, the network model is peer-to-peer. This method is a special kind of the polynomial based key generation scheme of Blundo et al. [26] which allows each pair of nodes to find a pairwise key. This method uses symmetric cryptography for key generation among the sensor nodes of the network. Blom's method can tolerate λ number of cryptanalyzed nodes without disturbing the security of the whole network.

Rahman et al. [27] proposed a key management scheme in order to improve Blom's method and at the same time keep its advantages. In this method, each pair of nodes in the network generate pairwise keys without exchanging messages and only by knowing each other's identifiers. This improvement reduces energy consumption, the memory and computations overload. Moreover, this method supports group key generation among sensor nodes in the network and provides the method with the possibility of adding new nodes to the network. This method works in both hierarchical and peer-to-peer network models and guarantees forward and backward secrecy and is resistant to different types of attacks including blackhole attacks, wormhole attacks, node capture attacks and denial-of-service attacks.

Erfani et al. [28] suggested a dynamic key management method which employs key pre-distribution and post-deployment mechanisms for key generation between pairs of nodes in the network. The network model in this method is peer-to-peer. In this key management method, the memory of each sensor node is divided into two parts. In the first of which the pre-distributed keys are stored and in the second part post-deployment keys are stored. Each pair of sensor nodes which are in the each other's radio range and share a common pre-distributed key, can have secure communication with each other. If two neighbour nodes share no common pre-distributed key, a post-deployment key is formed using key generation process. This method also supports rekeying process.

Hosen et al. [29] has presented a simple and powerful key management method with the introduction of a group mechanism in which the nodes of the network are divided into several small subgroups. In this method, key pre-distribution mechanism is used for the key generation among the network nodes and the required keys of the network are randomly chosen from a key source. This method reduces the need to store keys in each node and attempts to minimize the number of rekeying messages in the case of cryptanalysis of each node in the network, and consequently reduces communication overload of the network. The network model in this method is hierarchical.

Messai et al. [21] introduced energy aware hierarchical key management (EAHKM) method in wireless sensor networks. This method employs symmetric cryptography in the key generation process and uses both mechanisms of pre-distribution and post-deployment key generation. EAHKM has two phases: the pre-distribution phase, and the phase of key generation and cluster formation. In this method, before the development of the sensor nodes in the network, three pre-distributed keys are loaded in the memory of each node, then the cluster key and the pairwise key between each sensor node and the cluster head is generated in post-deployment manner. This method supports the processes of rekeying and adding new nodes to the network. Moreover, EAHKM is scalable, resilient and energy efficient and has proper resistance to cryptanalysis attacks.

Zhang et al. [19] presented hierarchical key management scheme (HKMS) for wireless sensor networks. In HKMS scheme, session keys are generated dynamically based on the number of hops in a cluster. To have secure communication within the cluster, the cluster key is employed. Moreover, before the development of sensor nodes in the network a global key is loaded in the memory of the sensor nodes which provides the key generation phase with secure communication and will be removed from the node memories after the development of the sensor nodes in the network. In the HKMS scheme, the data distribution range is delimited by the cluster head and the cluster size is delimited by the time to live (TTL) which is the number of hops in a cluster. If an adversary cryptanalyzes a sensor node in the network, with the cryptanalysis of the global key, all keys generated in the network will be cryptanalyzed. Therefore, HKMS does not have proper resilience against cryptanalysis attacks. Moreover, in the HKMS scheme, there is no provision of any mechanism for the movement of nodes in the network or for adding new nodes to the network.

Zho et al. [20] offered a key management method called localized encryption and authentication protocol (LEAP+). LEAP+ is a suitable scheme for the hierarchical wireless sensor networks. Furthermore, in this scheme, symmetric cryptography is employed in the key generation process. In LEAP+, four types of keys are stored in the memory of each node: individual key, pairwise key, cluster key, and a global key. Since in LEAP+ scheme, a global key is used for the communication between the nodes of the network, the security of the network could be threatened by the adversaries. If the global key is cryptanalyzed, the adversary can obtain the pairwise keys. Therefore, despite the scalability of LEAP+, it has little resilience and resistance to cryptanalysis attacks. Moreover, in LEAP+ there is no provision of a mechanism for adding new nodes to the network. LEAP+ scheme is effectively resistant to attacks such as HELLO flood attacks, node cloning attacks, and wormhole attacks.

A comprehensive and specific investigation of dynamic key management methods can be found in our previous study [14], where the above-mentioned key management schemes are discussed in detail and the pros and cons of each are presented. Table 1 provides a summary of the advantages and disadvantages of these key management methods. For a more thorough and detailed description, Ref. [14] is recommended.

In this paper, we propose a dynamic smart key management system (DSKMS) for wireless sensor networks. DSKMS design goals are: improving network security by removing cryptanalyzed nodes, increasing the resilience and resistance to cryptanalysis attacks, and reducing required memory and energy consumption in the network. The proposed system includes processes for key generation and distribution, key revocation and rekeying, and also for adding a new node and node movement inside the network.

3 Basic concepts

In this part, we will explain some of the basic concepts needed and used for our system.

3.1 Fuzzy logic

The theory of fuzzy logic was first introduced by Zadeh in 1965 in an essay titled “Fuzzy sets”. Fuzzy sets make partial membership possible. In other words, an element may partially belong to a set. Therefore, the results are not limited to absolutely true or absolutely false, but can be partially true or false [30].

Table 1 Comparison of advantages and disadvantages of key management methods

Method	Advantages	Disadvantages
Proposed scheme by Ahlawat et al. [22]	Acceptable security, simplicity, resistance against different types of attacks, low computation overload	High memory requirement, low scalability, adding new nodes to the network not feasible
μKMS [23]	Simplicity, feasibility of adding new nodes to the network, low energy consumption, low memory requirement, low computation overload	Low scalability
CL-EKM [24]	Acceptable security, support of node movement in the network, resistance against different types of attacks, feasibility of adding new nodes to the network	High memory requirement, low scalability, high energy consumption, high computation overload
Blom [25]	Acceptable security, resistance against different types of attacks, low energy consumption, low computation overload	Adding new nodes to the network not feasible, low scalability, high communication overload, high memory requirement
Proposed scheme by Rahman et al. [27]	Acceptable security, low communication and computation overload, resistance against different types of attacks, low energy consumption, feasibility of adding new nodes to the network	Low scalability, high memory requirement
Proposed scheme by Erfani et al. [28]	Simplicity, acceptable security, low energy consumption, low computation overload	Low scalability, adding new nodes to the network not feasible, high memory requirement
Proposed scheme by Hosen et al. [29]	Simplicity, low memory requirement, low energy consumption, low computation overload	Low resilience against attacks, adding new nodes to the network not feasible
EAHKM [21]	Suitable scalability, feasibility of adding new nodes to the network, low energy consumption, low computation overload	Use of global shared key among all network nodes, low resilience against attacks
HKMS [19]	Suitable scalability, low computation overload, low energy consumption, low memory requirement	Use of global shared key among all network nodes, low resilience against attacks, adding new nodes to the network not feasible
LEAP+ [20]	Suitable scalability, low computation overload	Use of global shared key among all network nodes, low resilience against attacks, adding new nodes to the network not feasible, high memory requirement, high communication overload

The main idea of the fuzzy logic includes "specialized experiences" of humans in the design phase and in the expression of the input-output process by means of a set of IF-THEN rules [30]. A fuzzy system includes the following parts which are illustrated in Fig. 1.

- Fuzzification: The process of changing the numerical data (crisp scale) to fuzzy values is called fuzzification which is performed through the fuzzy membership functions [31].
- Rule base: A fuzzy system is defined by a set of language sentences based on the knowledge of the specialists. This specialist knowledge is in the form of a set of IF-THEN rules which are simply applied to the fuzzy logic by fuzzy conditional sentences [31].
- Fuzzy inference: Fuzzy inference is the process of formulization of a nonlinear mapping from the input space to the output space. This mapping provides a basis for decision making. The process of fuzzy inference includes all the membership functions, operators, and IF-THEN rules [31].
- Defuzzification: Defuzzification is reversing of the fuzzification. Defuzzification is the process of turning a fuzzy value to a crisp value [31].

Different fuzzy systems employ different principles and methods for the aggregation of fuzzy rules. The most important mechanisms of fuzzy inference, which are widely employed in the fuzzy systems and applied programs, are Mamdani fuzzy inference and Sogeno (TSK) fuzzy inference. The difference between these two mechanisms, which are also called fuzzy models, is mainly in the areas of drawing conclusions from fuzzy rules, in aggregation, and in defuzzification processes. These two mechanisms are briefly explained below [30].

Mamdani fuzzy inference Mamdani fuzzy inference was for the first time projected by Mamdani and Assilian in 1974 using a set of fuzzy rules for the control of a steam engine and a steam tank. In Mamdani fuzzy model, crisp values are used for input. In this system, a kind of fuzzification is used in the input which turns crisp values to fuzzy values. There is also a defuzzification process in the output of this system which turns the output to a crisp value [31].

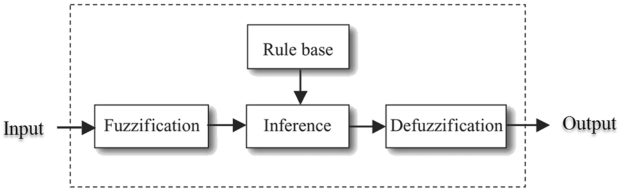


Fig. 1 Diagram of the architecture of fuzzy system [30]

Sogeno fuzzy inference Sogeno fuzzy inference, which is known as TSK, was suggested by Takagi, Sogeno, and Kang in 1985 and 1988 in order to develop a systematic method for generating fuzzy rules from the set of input-output data. The inputs and outputs of Sogeno fuzzy model are crisp values. The main problem with the Sogeno fuzzy model is that the "THEN" part of its fuzzy rule is a mathematic formula. Therefore, it doesn't provide a framework for the demonstration of human knowledge. This system cannot actualize different principles of fuzzy logic since it does not have the required flexibility to do so [31].

Fuzzy logic contributes to the smartification of applied programs and different systems. In the presented paper, we used Mamdani fuzzy inference mechanism in the process of generating path key and also for the purpose of determining the best cluster-head in the process of adding new nodes to the network. In this way, we tried to improve the security of the wireless sensor networks through the application of fuzzy logic.

3.2 Cryptography

Key cryptography methods can be divided into two categories: symmetric and asymmetric cryptography. These two processes are briefly explained below. In Fig. 2, cryptography process is demonstrated.

Symmetric key cryptography In symmetric key cryptography, the ciphering and deciphering keys used by the two nodes are the same [32]. Symmetric key cryptography does not guarantee a high level of security in the network, but is more efficient in terms of speed and energy consumption [33].

Asymmetric key cryptography If different keys are used for ciphering and deciphering processes, the algorithm of key cryptography is referred to as asymmetric [33]. Asymmetric key cryptography improves network security, but involves a high level of computation overload which results in an increase in energy and memory consumption of the sensor nodes [34].

In our system, we employed RC4 cryptography algorithm, which is a symmetric key cryptography, in the process of key generation between the sensor nodes of the network. This decision was based on the study of Boyle and Newe [35] which proved that in wireless sensor

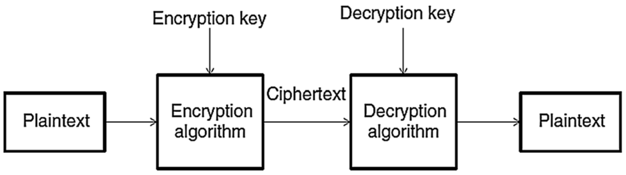


Fig. 2 Cryptography process [32]

networks, AES symmetric cryptography scheme consumes more energy than the RC4 symmetric algorithm and requires more time for encryption. Table 2 shows the juxtaposition of cryptography costs by RC4 algorithm and AES algorithm in the software and hardware. The size of each key code is supposed to be 128 bits [36].

4 Network model

The network model in our key management system (DSKMS) is heterogeneous and hierarchical wireless sensor network which includes a number of mobile sensor nodes, cluster-head nodes (CHs) and a base station (BS). This network is clustered through LEACH algorithm. In this network, the base station and the cluster-heads are immobile and other sensor nodes could be mobile. BS is in charge of managing and controlling the network and is equipped with an intrusion detection system (IDS). The cluster-head nodes receive the data collected by the sensor nodes and send them to the base station. The cluster-head nodes have higher levels of storage capacity, processing power, communication range, and energy than other sensor nodes. Each sensor node in the network has a unique identifier. The base station keeps a list of revoked nodes called list R which includes the ID of cryptanalyzed nodes. Whenever a node is cryptanalyzed, intrusion detection system (IDS) warns the base station, then BS adds the ID of cryptanalyzed node to the list R and removes it from the network. Figure 3 shows the network model of DSKMS as well as the nodes communication in the network.

5 The proposed key management system

In this part, the proposed dynamic smart key management system (DSKMS) for wireless sensor networks will be introduced. Our system includes the following processes:

The process of key generation and distribution This process is responsible for generating and distributing the keys used in the system which consists of the three following parts:

- The process of generating and distributing private key.
- The process of generating and distributing cluster key.

Table 2 Juxtaposition of cryptography costs of RC4 and AES algorithms [35]

	RC4	AES (software)	AES (hardware)
Encryption time (μs)	219.94	961.52	71.106
Encryption energy (mJ)	161	228	108

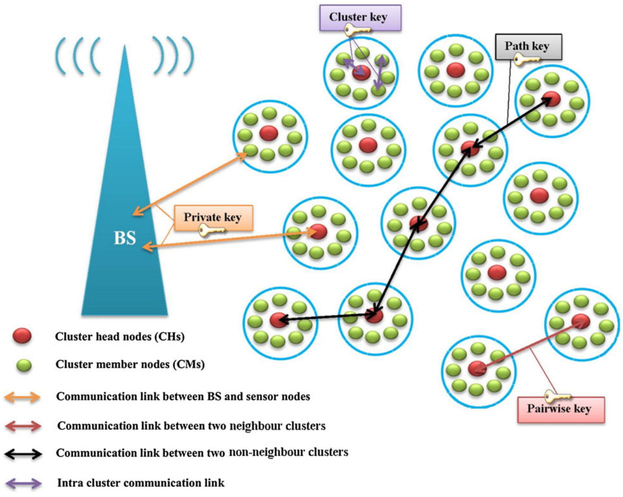

Cluster key
Path key
Private key
BS
Pairwise key
Cluster head nodes (CHs)
Cluster member nodes (CMs)
Communication link between BS and sensor nodes
Communication link between two neighbour clusters
Communication link between two non-neighbour clusters
Intra cluster communication link

Fig. 3 Network model in DSKMS and the communication among the network nodes

- The process of generating and distributing the key among the cluster-head nodes.

The process of key revocation and rekeying In dynamic key management systems, in order to neutralize the effect of the cryptanalyzed nodes on the whole network, provisions are made for the process of key revocation and rekeying. In our system, the process of key revocation and rekeying consists of the following two parts:

- The process of revoking and rekeying the cryptanalyzed cluster-head node.
- The process of revoking and rekeying the cryptanalyzed cluster member node.

The process of adding new node to the network Whenever a new node is to be added to the network, this process, taking several hops and using a fuzzy system, elects the best cluster as the new node, so that the this new node can be properly added to the network.

The process of node movement This process manages the movement, replacing and disabling sensor nodes in the network in order to maintain the security of the network.

Each of these processes will be explained in detail. In Table 3, a list of the symbols used in DSKMS are demonstrated.

5.1 The process of key generation and distribution

In DSKMS four types of keys are used to maintain the security of the communication links: private key, cluster key, pairwise key, and path key. The sensor nodes have a shared private key with the base station which use this key to communicate with BS. The cluster key is shared by all the nodes within a given cluster. The cluster member nodes

Table 3 The table of symbols

Symbol	Description
$E_k[M]$	Encryption of the message M with the key k
$D_k[M]$	Decryption of the message M with the key k
CM_i	Cluster member node i
CH_i	Cluster-head node i
$k_{Cluster\ key}$	Cluster key
$k_{Private\ key}$	Private key
$k_{Pairwise\ key}$	Pairwise key
$k_{Pairwise\ key_{CH_{Source},CH_i}}$	Pairwise key between CH_{Source} and CH_i
$k_{Path\ key}$	Path key
CH_{Source}	Source cluster-head node
$CH_{Destination}$	Destination cluster-head node
ID_{CH_i}	ID of the cluster-head i
$ID_{Cryptanalyzed\ Node}$	ID of the cryptanalyzed node
$ID_{Cryptanalyzed\ CH}$	ID of the cryptanalyzed cluster-head node
$ID_{Cryptanalyzed\ CM}$	ID of the cryptanalyzed cluster member node
R	List of revoked nodes
ID_n	ID of the new node n
N_i	The new node i

use the cluster key for the intra-cluster communication. This key is generated by the CH. Two neighbouring cluster-head nodes generate a pairwise key between themselves in order to make a secure communication link. If two non-neighbour cluster-heads are to have secure communication with each other, they make a path key between themselves. The process of generating and distributing these keys will be explained in detail.

5.1.1 The process of generating and distributing private key

Each sensor node has a private key for the communication with the base station. This key is generated through the key pre-distribution mechanism. Figure 4 shows how private keys are used. Each data packet which is exchanged between sensor nodes (cluster-heads or cluster member nodes) and the base station is coded via a private key and then transmitted. Before the deployment of the sensor nodes in the network, the base station randomly chooses a private key from the key source existing in the base station and allocates it to each sensor node of the network. Table 4 shows the key source of the base station. Algorithm 1 shows the process of private key generation and distribution and also data transmission between the sensor nodes and the base station in DSKMS.

```
Algorithm 1: The process of private key generation and distribution and data transmission between the sensor nodes and the base station.
Begin
  Private key generation and distribution
    Begin
      for i = 1 to n
        BS: Select a random number ($seed_i$) between 1 to n;
        BS: Derives private key from $seed_i$;
        BS: Send private key to sensor node;
      End
    End
  Data transmission between the BS and the sensor nodes
    Begin
      Sensor node (CHs or CMs): Compute $E_{Private\ key}[M]$ and send it to BS;
      BS: Compute $D_{Private\ key}[M]$ and get message M;
    End
End
```

5.1.2 The process of cluster key generation and distribution

The cluster member (CMs) nodes use the cluster key for intra-cluster communication. The cluster-head is responsible for the generation and distribution of this key among the cluster members. The cluster key is generated via the post-deployment key mechanism. Figure 5 shows the intra-cluster communication in system. If the sensor node i is to have a secure communication with the node j in the same

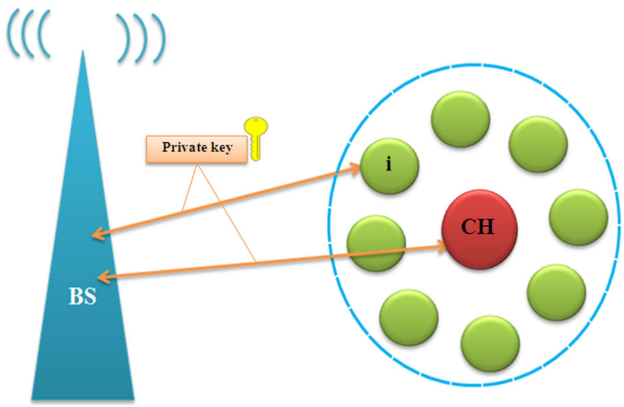


Fig. 4 The communication of the sensor nodes with the base station via private key

Table 4 The key source of the base station

Key source	
Seed	Private key
1	$k_{Private\ key_1}$
2	$k_{Private\ key_2}$
...	...
N	$k_{Private\ key_n}$

cluster (i.e. $CM_i \longrightarrow CM_j$), the data packets are coded with the cluster key and then transmitted. After the development of the cluster, the cluster head initiates the generation of the cluster key ($k_{Cluster\ key}$), then the cluster-head produces a message containing the cluster key and encrypts it via the private key of the cluster members ($k_{Private\ key}$) and sends it to its own cluster members. When the cluster members have received the message, they decode it using their private keys and then obtain the cluster key. Ultimately, an acknowledgement message is sent to the cluster-head node. The Algorithm 2 shows the process of cluster key generation and distribution and data transmission in a cluster in DSKMS.

Algorithm 2: The process of cluster key generation and distribution and data transmission in a cluster.

```
Begin
  Cluster key generation and distribution in a cluster
    Begin
      CH: Generate $k_{Cluster\ key}$;
      for i = 1 to $n_{cluster\ member}$
          CH: Compute $E_{Private\ key}[k_{Cluster\ key}]$ and send it to $CM_i$;
        $CM_i$: Compute $D_{Private\ key}[k_{Cluster\ key}]$ and get $k_{Cluster\ key}$;
        $CM_i$: Send $\ll Acknowledgement \gg$ to CH;
      End
    End
  Data transmission in a cluster between $CM_i$ , $CM_j$
    Begin
      IF $CM_i$ , $CM_j \in$ Same cluster THEN
        $CM_i$: Compute $E_{Cluster\ key}[M]$ and send it to $CM_j$;
        $CM_j$: Compute $D_{Cluster\ key}[M]$ and get message M;
      End
    End
End
```

5.1.3 The process of key generation and distribution among cluster-head nodes

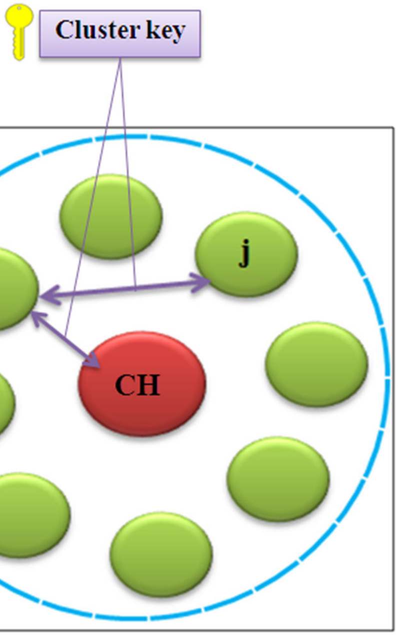


Fig. 5 Intra-cluster communication ($CM_i \longleftrightarrow CM_j$) or ($CM_i \longleftrightarrow CH$)

In DSKMS, whenever a member of the cluster is to communicate with a member of another cluster, this communication is done via the cluster-head nodes. In our system, the process of key production and distribution among the cluster-head nodes is of two modes:

- The two clusters are neighbours: If two cluster-head nodes are in the communication range of each other, we use the pairwise key generation mechanism to provide security for the communication and message exchange between them.
- The two clusters are not neighbours: If the two cluster-head nodes are not within the communication range of each other, we use the mechanism of path key generation to make a secure communication path between them and we enable them to exchange messages in a secure manner.

(a) *The mechanism of pairwise key generation and distribution between two neighbour clusters* When two clusters are in the same neighbourhood, their cluster-heads use pairwise key to have secure communication with each other. Figure 6 shows the pairwise key between two neighbour cluster-heads. Pairwise key is generated by the base station through post-deployment key mechanism. If the cluster-head i is to communicate with cluster-head j, which is its neighbour, (i.e. $CH_i \longrightarrow CH_j$), a pairwise key should be generated between them. For this purpose, at first, the cluster-head i sends a message to cluster-head j and asks for its identifier. This message is sent without coding because it does not contain sensitive information. After obtaining the ID of the cluster-head j, cluster-head i asks the base station to generate a pairwise key between the two cluster-heads which are CH_i and CH_j. The base station should first make sure that the two cluster-heads are valid, and for this purpose the base station searches the list R. This is done using the *Control_ID* function which is shown in Algorithm 3. If the identifiers of the cluster-head nodes are not present in the list R, the base station confirms the validity of these two nodes and makes a pairwise key between them and sends this key to each of these cluster-heads. The cluster-heads save this key in their memories so as to use it for exchanging messages between themselves. In Algorithm 3, the process of generation and distribution of pairwise key between neighbour cluster-head nodes in DSKMS is presented.

(b) *The mechanism of generating path key between two non-neighbour clusters* If two clusters are not neighbours, then source and destination cluster-heads are connected with each other through the intermediary cluster-heads. The intermediate cluster-head nodes function as a communication path between the source and destination cluster-head nodes. In this way, the path key among source and destination cluster-heads, which consists of a set of pairwise keys between the intermediate cluster-heads, is developed. The procedure of making the path key is performed through the suggested number one fuzzy smart decision-making system (FSDS1). To make the path key, it is necessary for the source cluster-head by FSDS1, elects a cluster-head from its neighbour cluster-heads ($CH_{neighbour}$) which is nearer to the destination cluster-head and which has more energy. For this purpose, the source cluster-head sends a message to its neighbouring cluster-heads enquiring their spatial coordinates ($x_{CH_{nei}}, y_{CH_{nei}}$) and their residual energy ($E_{residual}$). The information obtained from this enquiry is used for the calculation of input parameters of FSDS1. After electing the appropriate cluster-head by FSDS1, the pairwise key between the source cluster-head and the elected cluster-head is added to the pairwise keys contained in the path key. Then the elected cluster-head is assigned as the new source node, and this procedure is continued to the point of reaching the destination cluster-head ($CH_{destination}$). In Figure 9, the process of generating a path key between two non-neighbour cluster-heads via FSDS1 is presented.

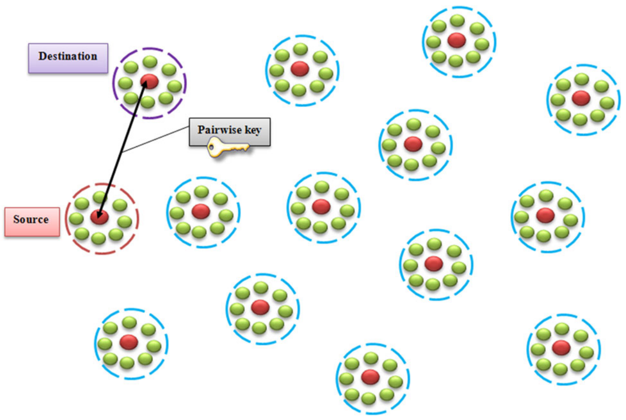


Fig. 6 Pairwise key between two neighbour cluster-head nodes

The input parameters in FSDS1 are:

- The distance between the neighbor cluster-head and the destination cluster-head (Distance): The distance from the neighbour cluster-head ($CH_{neighbour}$) to the destination cluster-head ($CH_{destination}$) is calculated on the basis of the Euclidean distance between these two cluster-heads via the Eq. 1:

$$Distance(CH_{neighbour}, CH_{destination}) = \sqrt{(x_{CH_{nei}} - x_{CH_{des}})^2 + (y_{CH_{nei}} - y_{CH_{des}})^2} \tag{1}$$

  In which, the $Distance(CH_{neighbour}, CH_{destination})$ is the Euclidian distance between $CH_{neighbour}$ and $CH_{destination}$. $(x_{CH_{nei}}, y_{CH_{nei}})$ are the spatial coordinates of the neighbour cluster-head node, and $(x_{CH_{des}}, y_{CH_{des}})$ are the spatial coordinates of destination cluster-head node.
- The energy of the neighbour cluster-head node (Energy): This parameter is obtained based on the residual energy of the neighbour cluster-head node $(E_{residual})$.

Algorithm 3: The process of generating and distributing pairwise key between neighbour cluster-head nodes.

```
Begin
  IF $CH_{Source}$ and $CH_{Destination}$ are neighbours THEN
    % Node discovery process.
    CH_Source: Send M ["Request node ID"] to $CH_{Destination}$;
    CH_Destination: Send $ID_{CH_{Destination}}$ to $CH_{Source}$;
    CH_Source: Send $ID_{CH_{Destination}}$ and $ID_{CH_{Source}}$ to the BS;
    %BS checks validity of two nodes. BS search list R
    for two nodes IDs by Control_ID function.
    %IF CH_i is validate THEN output of $Control\_ID(CH_i)$
    is 1.
    IF $Control\_ID(ID_{CH_{Source}}) = 1$ and
    $Control\_ID(ID_{CH_{Destination}}) = 1$ THEN
      BS: Generate $k_{Pairwise\ key}$;
      BS: Compute $E_{Private\ key}[k_{Pairwise\ key}, ID_{CH_{Destination}}]$ and send
      it to $CH_{Source}$;
      CH_Source: Compute $D_{Private\ key}[k_{Pairwise\ key}, ID_{CH_{Destination}}]$
      and get $k_{Pairwise\ key}$;
      BS: Compute $E_{Private\ key}[k_{Pairwise\ key}, ID_{CH_{Source}}]$ and send it
      to $CH_{Destination}$;
      CH_Destination: Compute $D_{Private\ key}[k_{Pairwise\ key}, ID_{CH_{Source}}]$
      and get $k_{Pairwise\ key}$;
    End
  End
End
```

In the design of FSDS1, Mamdani fuzzy inference is employed. In Mamdani fuzzy inference, a normalization procedure is needed for turning non-fuzzy parameters into fuzzy parameters. Therefore, the values of input parameters of FSDS1 should be normalized in the range of [0, 1]. The process of normalization is performed based on the Equation 2.

$$N_{norm} = \frac{N_{current}}{N_{max}} \tag{2}$$

In which, N_{norm} is the normalized value of the input parameter, $N_{current}$ is the current value of the input parameter, and N_{max} is the maximum value of the input parameter.

In Fig. 7 the fuzzy membership functions (FMFs) for the input parameters of distance and energy in FSDS1 are presented. Each input in the proposed fuzzy system is of three modes (low, medium, and high).

Figure 8 shows the fuzzy membership functions for the output parameter in FSDS1. The output includes 5 modes (very low, low, medium, high, and very high).

The set of rules employed in FSDS1 is shown in Table 5. For example, in Table 5, rule 1 is expressed as follows (Fig. 9).

Rule 1: IF Energy of the $CH_{neighbour}$ is *low* **AND** Distance between $CH_{neighbour}$ and $CH_{destination}$ is *low* **THEN** output is *medium*.

Algorithm 4 shows the process of path key generation in the inter-cluster communication in DSKMS.

The flow chart of the process of data transmission between two sensor nodes in DSKMS is shown in Fig. 10. As is shown in this figure, if the sensor node *i* is to send the message *M* to the sensor node *j*, the message *M* is coded and sent to the node *j* using the keys described in Sect. 5.1 which are the private key, cluster key, pairwise key and the path key.

Algorithm 4: The process of generating a path key in the inter-cluster communication.

```
Begin
  IF $CH_{Source}$ and $CH_{Destination}$ aren't neighbours THEN
    $k_{Path\ key} = \{\}$;
    $CH_i = CH_{Source}$
    While ($CH_i \neq CH_{Destination}$)
      FSDS1 (CHs);
      %FSDS1 output is best CH (CH_i).
      $k_{Path\ key} = k_{Path\ key} + \{k_{Pairwise\ key_{CH_{source},CH_i}}, ID_{CH_i}\}$;
      $CH_{Source} = CH_i$;
    End
  End
End
```

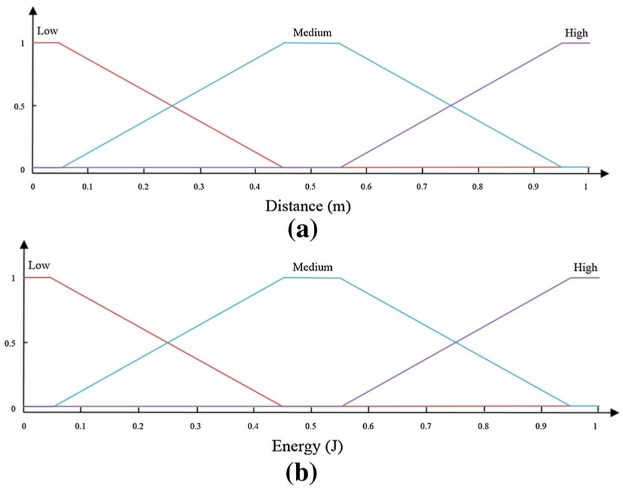


Fig. 7 Fuzzy membership function for the input parameters in FSDS1. **a** The distance between the neighbour cluster-head node and the destination node. **b** The residual energy of the neighbour cluster-head node)

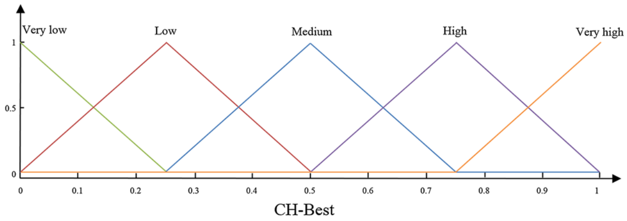


Fig. 8 Fuzzy membership functions for the output parameter in FSDS1

5.2 The process key revocation and rekeying

Rekeying is a very important part of any dynamic key management system. If the keys are not rekeyed in a long period of time, the probability of cryptanalysis of the network will increase. Therefore, if a cryptanalyzed node is identified, it is necessary to revoke the keys connected to it so as to keep the normal operation of the network. In DSKMS, the base station, using its intrusion detection system, has the ability to identify the cryptanalyzed nodes. Whenever the intrusion detection system detects a cryptanalysis attack to the network, it reports the ID of the cryptanalyzed node $(ID_{cryptanalyzed\ node})$ to the base station. The base station examines the ID of the cryptanalyzed node to see whether the cryptanalyzed node is a cluster-head node or a cluster member node. The process of key revocation and rekeying in DSKMS for both cases is explained below.

Table 5 The fuzzy rules for FSDS1

Fuzzy rules	Energy of the $CH_{neighbour}$	Distance between $CH_{neighbour}$ and $CH_{destination}$	Output
1	Low	Low	Medium
2	Low	Medium	Low
3	Low	High	Very low
4	Medium	Low	High
5	Medium	Medium	Medium
6	Medium	High	Low
7	High	Low	Very high
8	High	Medium	High
9	High	High	Medium

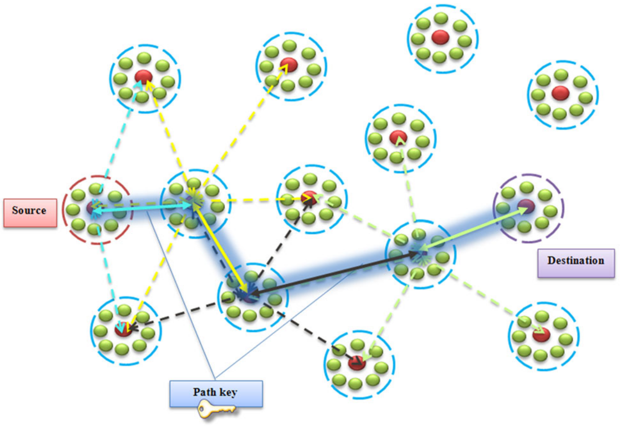


Fig. 9 The generation of path key between two non-neighbour cluster-heads using FSDS1

5.2.1 The process of key revocation and rekeying the cryptanalyzed cluster-head node

If a cluster-head node is cryptanalyzed, its private key, pairwise keys with other cluster-head nodes and its cluster key will be cryptanalyzed. As shown in Fig. 11, whenever the cluster-head node is cryptanalyzed, the intrusion detection system sends a warning and reports the ID of the cryptanalyzed cluster-head $(ID_{cryptanalyzed\ CH})$ to the base station. The base station adds the ID of the cryptanalyzed cluster-head to the list R and sends a message containing the ID of the cryptanalyzed cluster-head to the member nodes of that cluster and to the other cluster-head nodes connected to the cryptanalyzed cluster-head, so that these nodes revoke the keys connected to the cryptanalyzed cluster-head.

Moreover, whenever this message is received by other cluster-head nodes, they revoke their pairwise key with the cryptanalyzed cluster-head node. Since the ID of the cryptanalyzed cluster-head is included in list R, if this cluster-head node requests other cluster-heads to form new pairwise key, the base station examines the validity of this node and finds it illegal and the request is denied.

Furthermore, when the cluster member nodes receive this message from the base station, they revoke their cluster key and start the process of joining to a new cluster. Since the ID of the cryptanalyzed cluster-head node is registered in list R, if this cluster-head attempts to form a new cluster, it cannot accept new nodes. This process is shown in Fig. 12.

5.2.2 The process of key revocation and rekeying the cryptanalyzed cluster member node

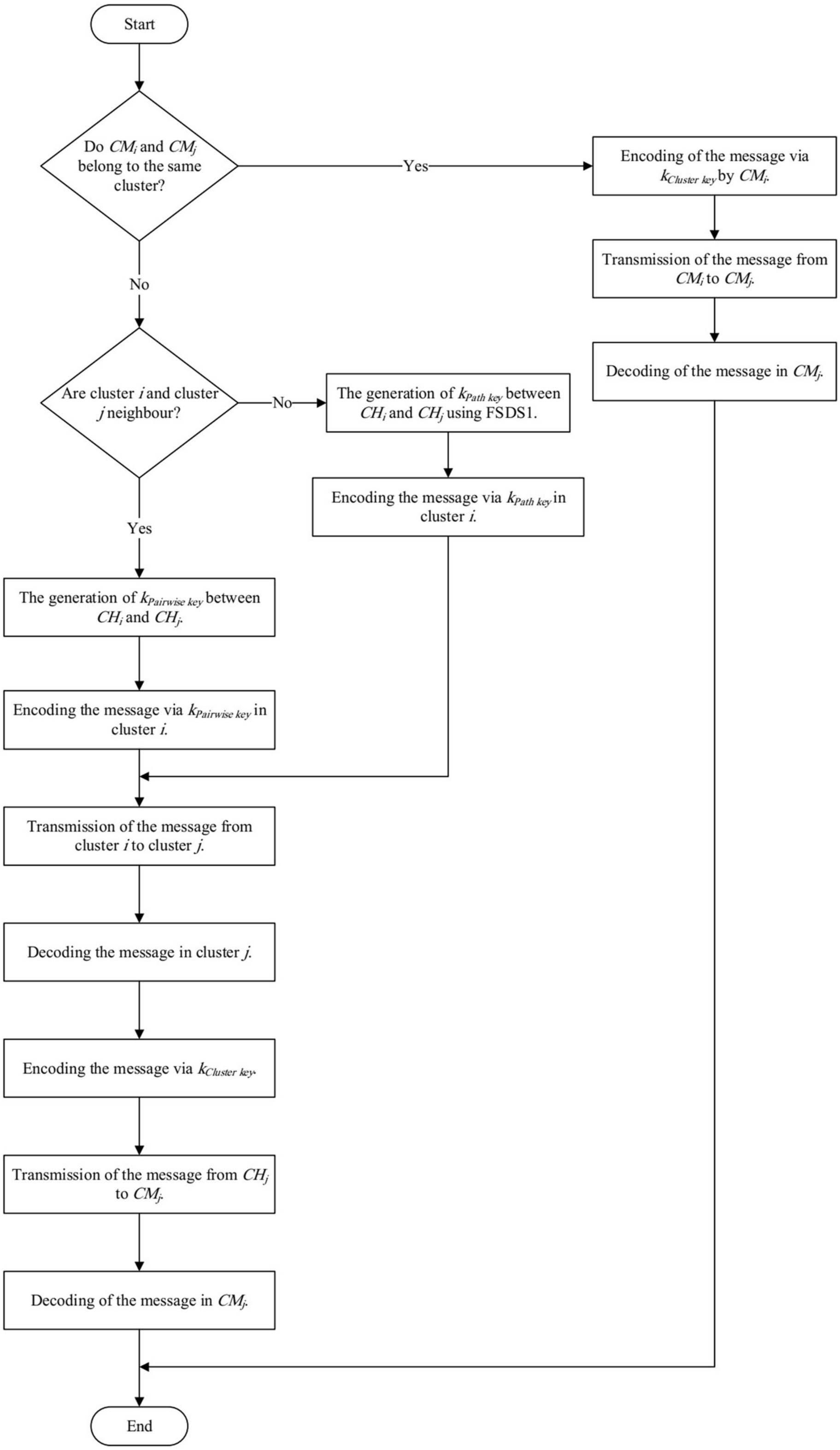
Whenever a cluster member node is attacked and cryptanalyzed, the intrusion detection system sends a warning containing the ID of the cryptanalyzed cluster member node $(ID_{cryptanalyzed\ CM})$ to the base station. The base station adds the ID of the cryptanalyzed cluster member node to the list R and sends a message containing the ID of the cryptanalyzed cluster member node to the cluster-head node connected to this node so that the cluster-head would rekey its cluster key. When this message is received, the cluster-head node generates the new cluster key and sends it to all its members except the cryptanalyzed node. In this way, the cryptanalyzed cluster member node would be unable to send messages in this cluster and also would be unable to connect to a new cluster because the ID of the cryptanalyzed node is registered in list R. This process is shown in Fig. 13.

Algorithm 5 shows the process of key revocation and rekeying in DSKMS, in case of a sensor node being cryptanalyzed. The flow chart of this process is also presented in Fig. 14.

5.3 The process of adding a new node to the network

When a new node is to be added to the network, at first the base station will examine the validity of this node. If the ID of this node is not registered in list R, then the base station would assign a private key to this node. Then this node will broadcast a join message to the cluster-head nodes of the

Fig. 10 The flow chart of the process of data transmission between the two sensor nodes CM_i and CM_j in DSKMS


Start
Do CM_i and CM_j belong to the same cluster?
Yes
Encoding of the message via $k_{Cluster\ key}$ by CM_i.
Transmission of the message from CM_i to CM_j.
Decoding of the message in CM_j.
No
Are cluster i and cluster j neighbour?
No
The generation of $k_{Path\ key}$ between CH_i and CH_j using FSDS1.
Encoding the message via $k_{Path\ key}$ in cluster i.
Yes
The generation of $k_{Pairwise\ key}$ between CH_i and CH_j.
Encoding the message via $k_{Pairwise\ key}$ in cluster i.
Transmission of the message from cluster i to cluster j.
Decoding the message in cluster j.
Encoding the message via $k_{Cluster\ key}$.
Transmission of the message from CH_j to CM_j.
Decoding of the message in CM_j.
End

network. Furthermore, the new node sending a message to the cluster-head nodes enquires their spatial coordinates (x_{CH_i}, y_{CH_i}) and the number of nodes in each cluster so as to use the obtained information for the calculation of the input parameters of the presented number two fuzzy smart decision-making system (FSDS2). FSDS2 elects the best cluster-head from the cluster-head nodes in order to add the new node to the members of that cluster. In the design of

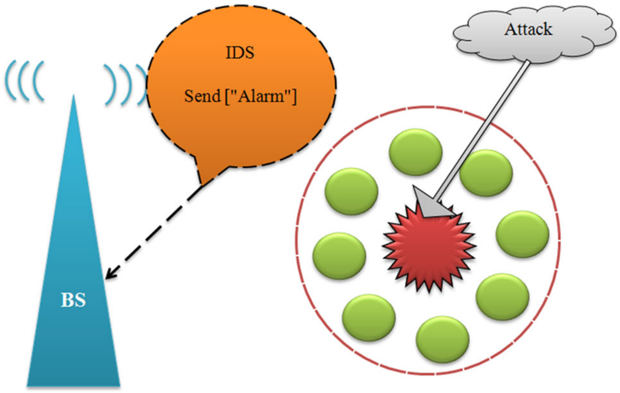


Fig. 11 The identification of the cryptanalyzed cluster-head node by the intrusion detection system

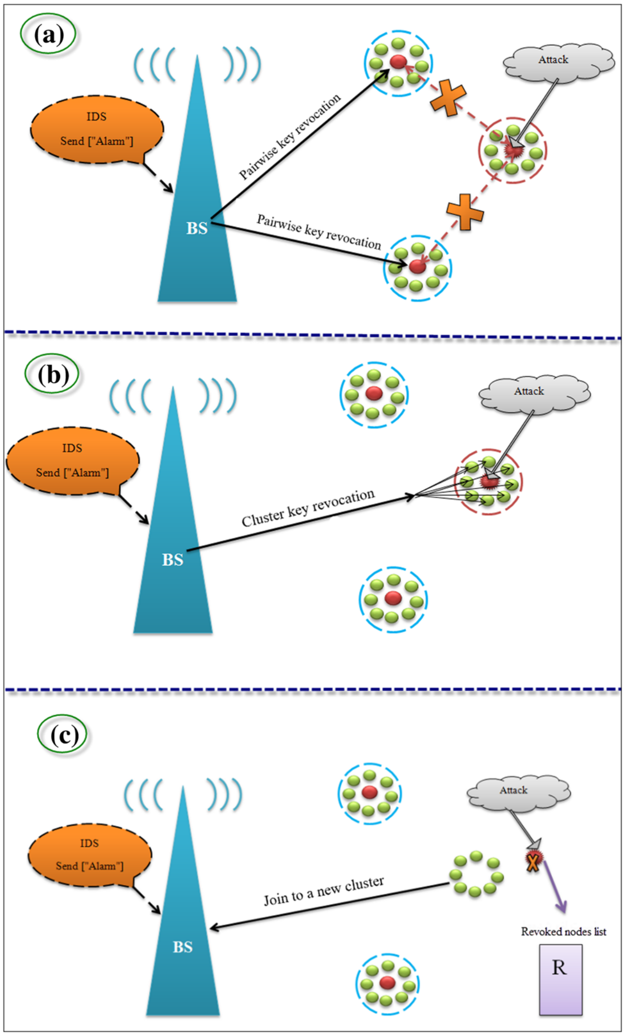


Fig. 12 The process of rekeying in the case of a cluster-head node being cryptanalyzed. **a** Removing pairwise keys by neighbour cluster-heads, **b** cluster key revocation, **c** the request by the cluster member nodes to join to a new cluster

FSDS2, we used the Mamdani inference mechanism for the process of adding new nodes to the network. The input parameters in FSDS2 are the followings:

- The distance of the new node from the cluster-head (Distance): The distance of the new node to each of the cluster-heads is calculated based on the Euclidian distance between the two via Eq. 3.

$$Distance(CH_i, SN_{new}) = \sqrt{(x_{CH_i} - x_{SN_{new}})^2 + (y_{CH_i} - y_{SN_{new}})^2} \tag{3}$$

  In which, Distance (CH_i, SN_{new}) is the Euclidian distance between the new node (SN_{new}) and the cluster-head i (CH_i), in a way that $1 \leq i \leq d$ in which d is the total number of the cluster-heads which have received the join message of the new node. Moreover, (x_{CH_i}, y_{CH_i}) is the spatial coordinates of the cluster-head i (CH_i) and $(x_{SN_{new}}, y_{SN_{new}})$ is the spatial coordinates of the new node (SN_{new}).
- The number of nodes in each cluster: The number of nodes in a cluster should not transcend a specific threshold since increasing the number of the nodes in a cluster results in the reduction of the data transmission rate and an increase in latency and energy consumption in the network. This threshold will be determined proportionately to the number of nodes existing in the network. The cluster-head is aware of the number of nodes within its own cluster.

The input parameters of FSDS2 are normalized by means of the Eq. 2 which was introduced in Sect. 5.1.3. The fuzzy membership functions for the input parameters of FSDS2 are shown in Fig. 15. Each input parameter consists of three modes (low, medium, and high).

The fuzzy membership functions for the output parameter of FSDS2 are shown in Fig. 16. The output includes 5 modes (very low, low, medium, high, and very high). Moreover, FSDS2 follows the fuzzy rules presented in

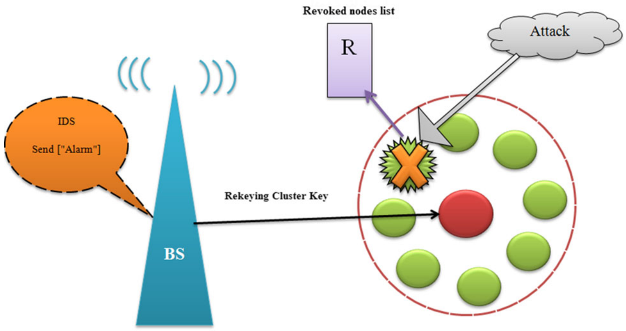


Fig. 13 The process of rekeying in the case of a cluster member node being cryptanalyzed

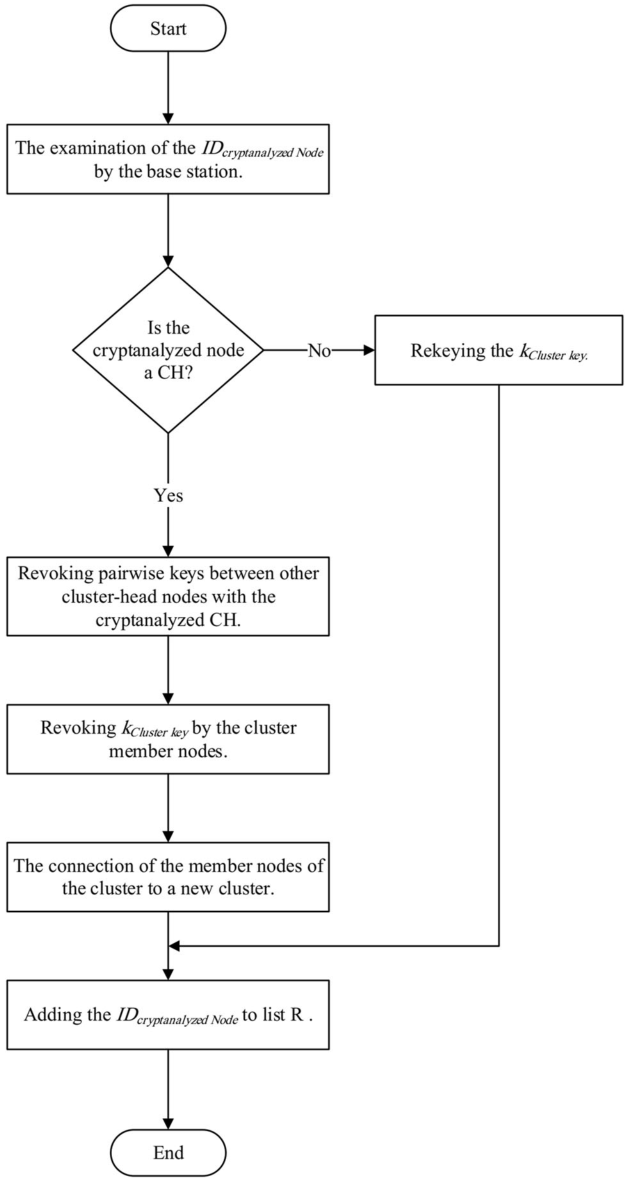

Start
The examination of the $ID_{cryptanalyzed\ Node}$ by the base station.
Is the cryptanalyzed node a CH?
No
Rekeying the $k_{Cluster\ key.}$
Yes
Revoking pairwise keys between other cluster-head nodes with the cryptanalyzed CH.
Revoking $k_{Cluster\ key}$ by the cluster member nodes.
The connection of the member nodes of the cluster to a new cluster.
Adding the $ID_{cryptanalyzed\ Node}$ to list R .
End

Fig. 14 The flow chart of the process of key revocation and rekeying in DSKMS

Table 6. For example, in Table 6, rule 1 is expressed as follows.

Rule 1: I F Number of nodes in a cluster is *low* **AND** Distance of the new node from CH is *low* **THEN** output is *very high*.

In Algorithm 6, the process of adding a new node to the network in DSKMS is presented. The flow chart of this process is also shown in Fig. 17.

5.4 The process of node movement

A node may leave a cluster because it has changed its location or because of the internal disconnection. In our system, there are two modes for identifying a node leaving a cluster:

Algorithm 5: The process of key revocation and rekeying.

```
Begin
  IF (IDS: Send M ["Alarm for attack", $ID_{cryptanalyzed\ Node}$]
  to the BS) THEN
  %Check ID of the cryptanalyzed node.
    IF $ID_{cryptanalyzed\ Node} \in CH$ THEN
      BS: Compute $E_{Private\ Key}$ (M ["Revoke pairwise
      key with cryptanalyzed CH", $ID_{cryptanalyzed\ CH}$]) and
      send it to CHs;
      CHs: Compute $D_{Private\ Key}$ (M ["Revoke pairwise
      key with cryptanalyzed CH", $ID_{cryptanalyzed\ CH}$]);
      CHs: Revoke pairwise key with cryptanalyzed CH;
      BS: Compute $E_{Private\ Key}$ (M ["Revoke cluster key
      with cryptanalyzed CH", $ID_{cryptanalyzed\ CH}$]) and send
      it to CMs;
      CMs: Compute $D_{Private\ Key}$ (M ["Revoke cluster
      key with cryptanalyzed CH", $ID_{cryptanalyzed\ CH}$]);
      CMs: Revoke cluster key with cryptanalyzed CH;
      BS: Register $ID_{cryptanalyzed\ CH}$ in list R;
      CMs: Compute $E_{Private\ Key}$ (M ["Join to a new CH"])
       and send it to BS;
    End
    IF $ID_{cryptanalyzed\ Node} \in CM$ THEN
      BS: Compute $E_{Private\ Key}$ (M ["Rekeying cluster key",
      $ID_{cryptanalyzed\ CM}$]) and send it to CH;
      CH: Compute $D_{Private\ Key}$ (M ["Rekeying cluster key",
      $ID_{cryptanalyzed\ CM}$]);
      BS: Register $ID_{cryptanalyzed\ CM}$ in list R;
      CH: Rekeying cluster key;
    End
  End
End
```

- Mode 1: The cluster member node decides to leave the cluster and sending a message informs the cluster-head. When the cluster-head receives the message of node movement, it rekeys its cluster key and the mobile node performs the process of adding new node to the network which was already described in Sect. 5.3, so as to join a new cluster.
- Mode 2: The cluster-head has a problem communicating with a member node of its own cluster. This incident may happen because the battery energy of the node is finished, or the node is being the target of a cryptanalysis attack, or because of an unwanted movement of the node. The member nodes of a cluster periodically send a *Beacon message* to their cluster-head. If after a given period of time, the cluster-head does not receive the *Beacon message* from a member node, it diagnoses an unwanted movement of the node from its cluster and rekeys the cluster key. In Algorithm 7, the process of node movement in DSKMS is

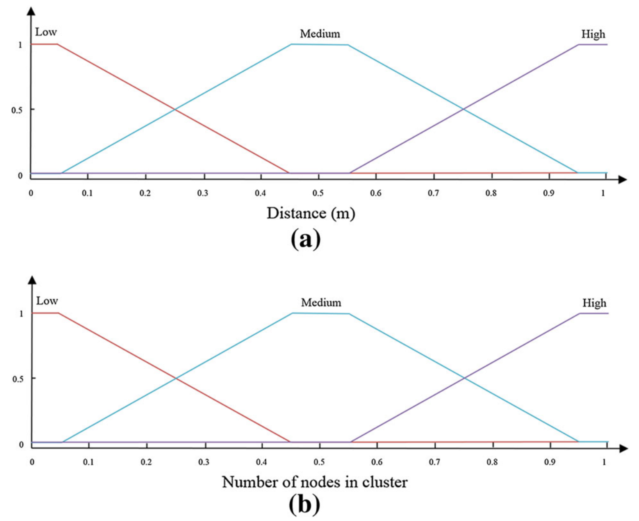


Fig. 15 The fuzzy membership functions for the input parameters in FSDS2. **a** The distance of the new node from the cluster-head nodes, **b** the number of nodes contained in a cluster

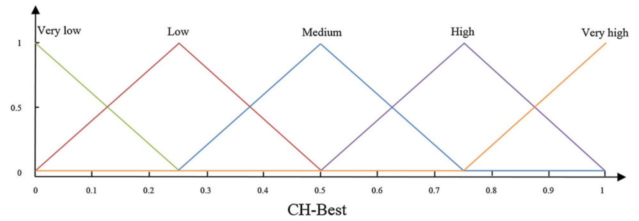


Fig. 16 The fuzzy membership functions for the output parameter of FSDS2

presented. The flow chart of this process is also presented in Fig. 18.

6 Performance analysis

In this part, the security performance of DSKMS will be examined using security scales like forward and backward secrecy, resilience and resistance to attacks.

6.1 Forward and backward secrecy

Forward secrecy guarantees that the passive adversary cannot use the old keys for decoding the new messages [37]. Backward secrecy guarantees that the passive adversary using new keys cannot decode old messages [38].

DSKMS performs rekeying in the process of node movement, in the process of adding new nodes to the network and also when a node is cryptanalyzed. Whenever a node is removed from the network, all of its keys become invalid and the cluster-key assigned to it will be rekeyed. The cluster-head send the new cluster key to all the nodes with the exception of the revoked node. Therefore, this revoked node cannot decode a new message using its old cluster key. Moreover, whenever a node is added to the cluster, the cluster-head generates a new cluster key and sends it to all nodes that are members of the cluster. Therefore, the new node cannot decode old messages via the new cluster key. Therefore, DSKMS guarantees forward and backward secrecy.

Algorithm 6: The process of adding a new node to the network.

```
Begin
  BS: Assigning an ID to the new node ($N_i$);
  %BS checks validity of the node. BS search list R for
  node ID by Control_ID function. IF node ID is validate
  THEN return 1.
  IF $Control\_ID\ (ID_n) = 1$ THEN
    BS: Assigning a private key to the new node ($N_i$);
  End
  %The process of adding the new node to a cluster.
  $N_i$: Broadcast M ["Join to a cluster"] to CHs;
  Step(1): FSDS2 (CHs);
  %FSDS2 output is best CH ($CH_i$).
  $CH_i$: Send M ["Accept this node"] to $N_i$;
  IF $Control\_ID\ (CH_i) = 1$ THEN
    $N_i$: Send $\ll$Acknowledgement$\gg$ to $CH_i$;
    $CH_i$: Rekeying cluster key;
  ELSE
      Go to Step(1);
  End
End
```

Algorithm 7: The process of node movement.

```
Begin
  IF CH doesn't receive beacon message from $CM_i$ THEN
    CH: Rekeying cluster key;
  End
  IF $CM_i$: Send M ["Leave cluster"] to CH THEN
    CH: Rekeying cluster key;
    $CM_i$: Broadcast M ["Join to a cluster"] to CHs;
  End
End
```

Table 6 The fuzzy rules for FSDS2

Fuzzy rules	Number of nodes in a cluster	Distance of the new node from CH	Output
1	Low	Low	Very high
2	Low	Medium	High
3	Low	High	Medium
4	Medium	Low	High
5	Medium	Medium	Medium
6	Medium	High	Low
7	High	Low	Medium
8	High	Medium	Low
9	High	High	Very low

6.2 Resistance to attacks

Resistance to attacks is considered a security scale for the evaluation of key management schemes of wireless sensor networks [39]. A proper dynamic key management scheme should have the ability to resist attacks to the network so that the network can continue its normal operation [40].

In our key management system (DSKMS), if an adversary cryptanalyzes a cluster member node, the cluster key assigned to this node will be revealed. But the cryptanalysis of this key will not have any effect on the operation of other clusters and with the rekeying of the cluster key, the cryptanalyzed node will be removed from the network. Moreover, if the cluster-head node is cryptanalyzed, no harm will be done to the security of the communications of other clusters, because the security of the communications between neighbour cluster-heads is maintained by means of pairwise keys and when the pairwise keys of the cryptanalyzed cluster-head with the neighbour cluster-heads are revealed, the adversary will remain uninformed of the pairwise keys between other cluster-heads and the security of the communications will be maintained. Furthermore, the mechanism of path key generation between two non-neighbour cluster-heads increases the resistance of the network to the attacks. Consequently, DSKMS effectively guarantees resistance to attacks.

6.3 Resilience

Resilience refers to the resistance of network against cryptanalysis of a node. If the adversary cannot affect any node except the cryptanalyzed node, then the resilience of key management system is high [41]. On the other hand, if the cryptanalysis of a node results in the cryptanalysis of the whole network, then the resilience of key management system is low [42].

Since DSKMS is capable of key revocation and rekeying process, it quickly removes cryptanalyzed node from the network and by registering this node in the list R, prevents this node from membership in other clusters and consequently, prevents it from affecting other nodes of the network. Therefore, we can claim high resilience for DSKMS.

7 Simulation and results evaluation

In this part, to evaluate the functioning of DSKMS, we implemented it in the simulator software NS-Alinone-2.35. We here will compare the results of this simulation with simulation results of three other key management schemes of HKMS [19], LEAP+ [20] and EAHKM [21].

To do the simulation, we supposed that in the wireless sensor network, 250 sensor nodes are randomly scattered in an area of $2500 \times 2500\,\mathrm{m}^2$. In this network, the base station is located in the center and is immobile. The number of cluster-heads is "seven percent" of the number of the nodes in the network which are scattered randomly in the network. The cluster-heads are immobile. Other sensor nodes which are scattered randomly in the network could be mobile. The speed of the movement of the sensor nodes varies from 10 to 50 m/s. The speed, time of movement and the direction of movement of the sensor nodes are randomly chosen. To improve the accuracy of the results of the simulation, we have repeated the simulation operation 25 times.

The simulation parameters are listed in Table 7. The simulation results of DSKMS will be compared with the results of key management schemes of HKMS [19], LEAP+ [20] and EAHKM [21], in terms of communication overload, consumed energy, the required memory space, and the effect of the cryptanalyzed nodes on the network communication.

7.1 Communication overload

Since in key management schemes the control messages exchanged among the sensor nodes are responsible for most energy consumption in the network, to evaluate the

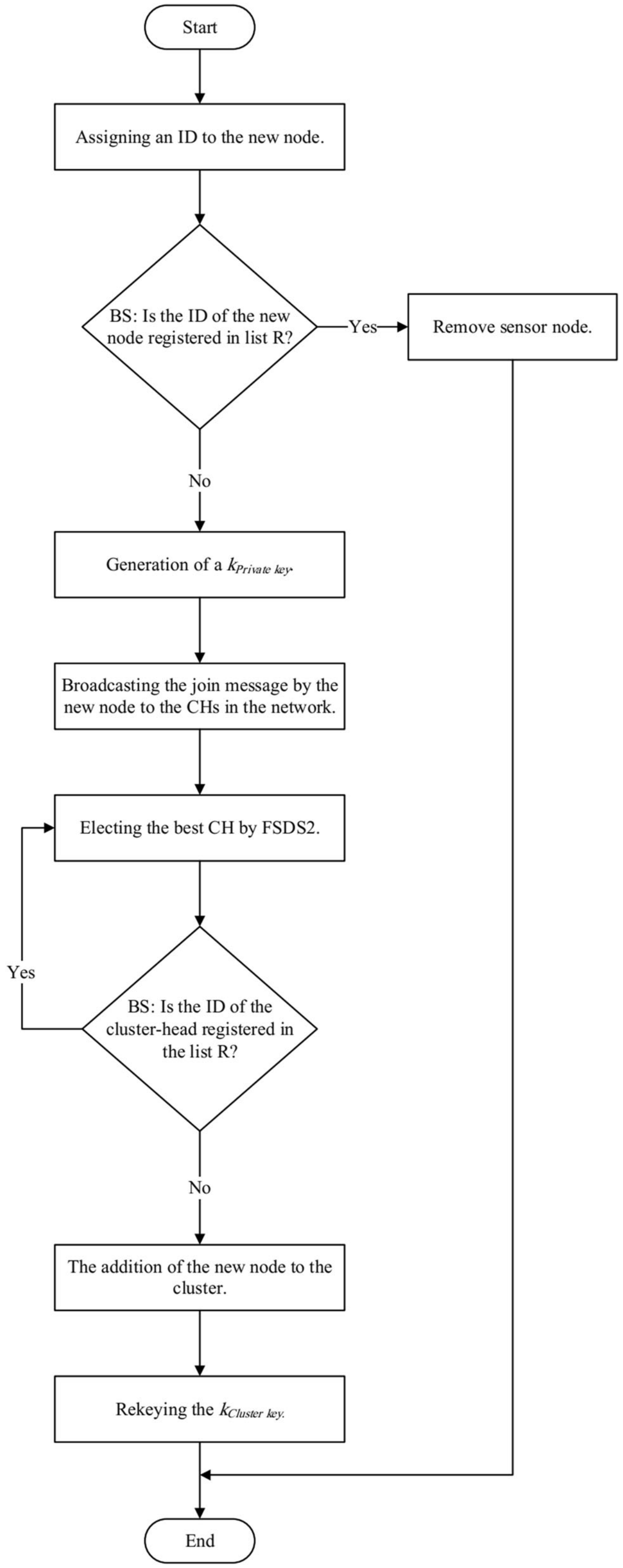

Start
Assigning an ID to the new node.
BS: Is the ID of the new node registered in list R?
Yes
Remove sensor node.
No
Generation of a $k_{Private\ key}$.
Broadcasting the join message by the new node to the CHs in the network.
Electing the best CH by FSDS2.
Yes
BS: Is the ID of the cluster-head registered in the list R?
No
The addition of the new node to the cluster.
Rekeying the $k_{Cluster\ key}$.
End

Fig. 17 The flow chart of the process of adding a new node to the network in DSKMS

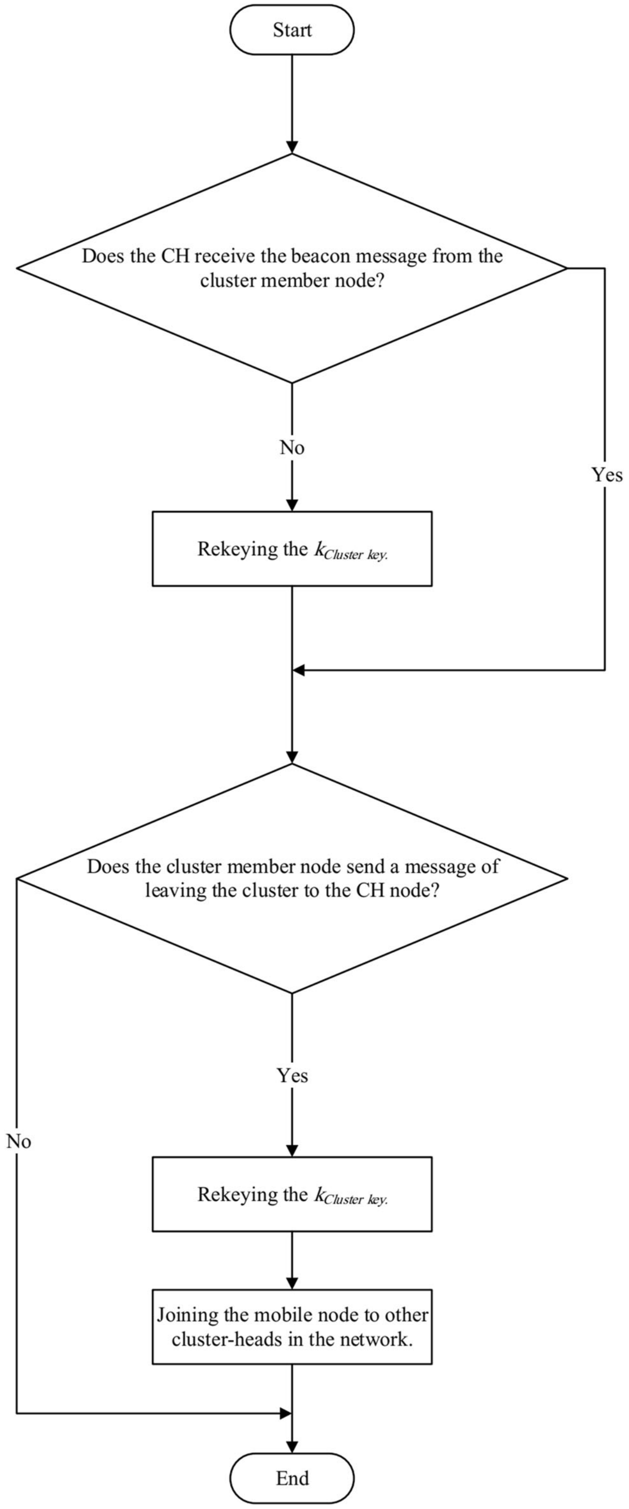

Start
Does the CH receive the beacon message from the cluster member node?
No
Yes
Rekeying the $k_{Cluster\ key}$.
Does the cluster member node send a message of leaving the cluster to the CH node?
Yes
No
Rekeying the $k_{Cluster\ key}$.
Joining the mobile node to other cluster-heads in the network.
End

Fig. 18 The flow chart of the process node movement in DSKMS

communication overload, the number of control messages received by the sensor nodes should be calculated. Obviously, if the communication overload in a key management system is high, this system consumes too much energy for maintaining secure communication among the sensor nodes of the network and consequently the sensor nodes quickly die. As a result of this, the lifetime of the network will be shortened. In Fig. 19, we have compared the communication overload of DSKMS with those of the three other key management schemes of HKMS [19], LEAP+ [20], EAHKM [21] in various network sizes which take into account the average number of received control messages by each sensor node in the process of cluster key generation. As seen in Fig. 19, in DSKMS the communication overload is lower than those of the other schemes. In DSKMS, in the process of adding new node to the network, the new sensor node broadcasts a message to the cluster-head nodes. When this message is received by the cluster-head nodes, FSDS2 elects the most suitable cluster-head node among the cluster-head nodes. Then the sensor node joins to that cluster and ultimately the new cluster key is generated by the cluster-head and sent to the member nodes of the cluster.

Suppose a sensor node is to connect to a cluster, and d cluster-head nodes are in the neighbourhood of this sensor node, the communication overload is calculated in the following way: a message is broadcasted by the sensor node to the cluster-head nodes for joining to a cluster, and this node will receive at most d messages from neighbour cluster-heads. But since DSKMS uses FSDS2 in the process of adding new node to the network, the number of messages sent by the cluster-head nodes may be lower than d. The elected cluster-head also sends a message containing the cluster key to this node. As a result, the communication overload in DSKMS is lower than or equal to $d + 2$. In EAHKM [21] scheme, each sensor node broadcasts a message and receives d messages from its neighbours. Then, the cluster-head sends a message containing its cluster key to this node. Therefore, in this scheme the communication overload is $d + 2$ messages. In LEAP+ [20] scheme, each sensor node broadcasts a message and receives d messages from its neighbours, then for cluster key generation sends d messages to its neighbours. Therefore, the communication overload of this system is $2d + 1$. In HKMS [19] scheme, each cluster-head sends a number of messages which depends on the TTL value and receives d messages from its neighbours. These results are summarily shown in Table 8.

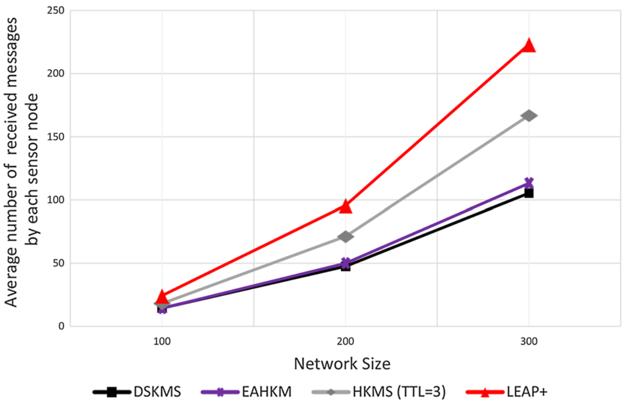


Fig. 19 The comparison of the communication overload of DSKMS with other schemes

7.2 The required memory space

Figure 20 shows the total memory space required for storing the keys for DSKMS compared with those of three key management schemes of HKMS [19], LEAP+ [20], and EAHKM [21] in different network sizes. As seen in Fig. 20, the memory space requirements of DSKMS is near to that of HKMS [19], and that of EAHKM [21] and is much lower than that of LEAP+ [20]. In DSKMS, the sensor nodes contain two keys in their memories: the private key for communication with the base station, and the cluster key for intra-cluster communications. The cluster-head nodes, in addition to these two keys, use pairwise keys or path keys for communication with other clusters. In EAHKM [21] scheme, the sensor nodes store three pre-distribution keys in their memories and in addition to these, use a cluster key for intra-cluster communications and pairwise keys for communication with the cluster-head. In

Table 7 Simulation parameters

Network size	2500 m× 2500 m
The location of the base station	The center of network (1250 m × 1250 m)
The number of network nodes	250
The number of cluster-head nodes	17 (7% of network nodes)
The number sensor nodes	233
The initial energy of cluster-head nodes	100 J
The initial energy of sensor nodes	10 J
Antenna	OminiAntenna
Key size	128 bit
Mac layer protocol	Mac/802-11

Table 8 The comparison of the communication overload of different schemes

Schemes	Communication overload
DSKMS	$\alpha \leq d + 2$
EAHKM [21]	$\alpha = d + 2$
LEAP+ [20]	$\alpha = 2d + 1$
HKMS [19]	Depends on the TTL value

d: Cluster-head nodes, α: Communication overload

HKMS [19] scheme, an initial key is stored in the memory of sensor nodes in a pre-distributed manner. Moreover, in this scheme, session keys are generated between sensor nodes, and a cluster key is generated between the cluster-head and the cluster members and all these are stored in the memory of the nodes. The memory space requirements for storing the keys in LEAP+ [20] is very high, since this scheme makes use of individual key, global key, pairwise keys with the neighbours, and cluster keys.

7.3 Energy consumption

Because of the limitedness of the energy of sensor nodes, the key management schemes should be energy-efficient. Figure 21 shows the average residual energy of the sensor nodes in network after the process of key generation. As seen in Fig. 21, DSKMS is optimal in terms of energy consumption. This is because in DSKMS we used FSDS1 in the process of path key generation and used FSDS2 in the process of adding new nodes to the network. The FSDS1 contributes to the reduction in energy consumption by generating the optimal path key between two non-neighbour cluster-heads. The fuzzy system FSDS2 also helps the sensor nodes to connect to the most suitable cluster which are of the shortest distance to the cluster-head and prevents the number of the nodes in a cluster to exceed a given threshold. Therefore, the fuzzy system FSDS2 prevents the nodes density within a cluster and in this way contributes to optimal energy consumption. As a result, DSKMS contributes to a longer network lifetime by means of saving energy in the process of key generation in sensor nodes. In HKMS [19] scheme, the member nodes of the cluster send their messages by multi-hop method to their cluster-heads and therefore have higher energy consumption in comparison to the single-hop transmission. In EAHKM [21] scheme, the sensor nodes among themselves elect the node with the highest residual energy as their cluster-head. The cluster-heads perform the operations of coding and decoding and receive more messages than member nodes of the cluster do. In this way, the lifetime of low energy nodes is elongated and consequently the lifetime of the network is enhanced. The LEAP+ [20] scheme, has the highest energy consumption compared with other schemes, since the sensor nodes in this scheme form

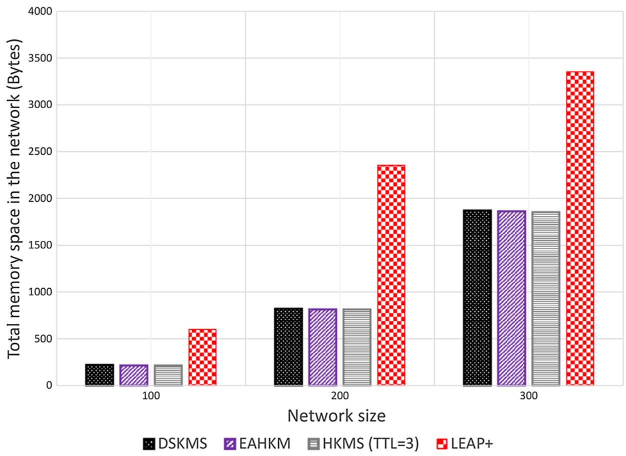


Fig. 20 The comparison of the total memory requirements for storing keys in DSKMS and other schemes

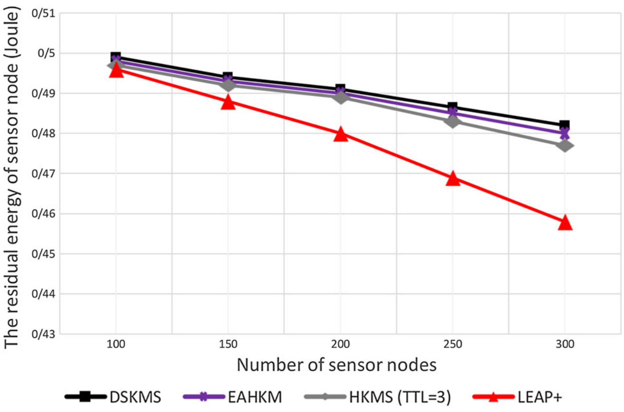


Fig. 21 The comparison of the residual energy of sensor nodes of the network after the process of key generation in DSKMS and in other schemes

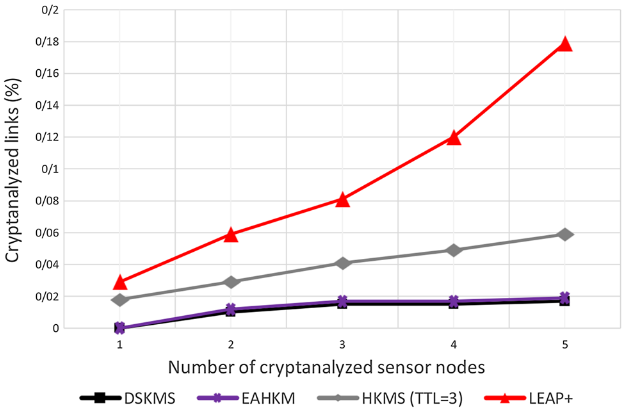


Fig. 22 The comparison of the resilience of DSKMS and other schemes

pairwise keys to communicate with each other and the number of control messages in this scheme is very high.

7.4 Resilience

In Fig. 22 the resilience of DSKMS is compared to those of the key management schemes of HKMS [19], LEAP+ [20], and EAHKM [21] against the cryptanalysis attacks, when an adversary randomly cryptanalyzes 1 to 5 sensor nodes. As shown in Fig. 22, DSKMS is more effective than the schemes of HKMS [19], LEAP+ [20] and the functions almost similar to EAHKM [21] scheme. In DSKMS, if a node is cryptanalyzed it would not affect other communications of the network and the cryptanalyzed node will be quickly removed from the network.

8 Conclusion

Security is one of the most important challenges of the wireless sensor networks. In the presented paper, we proposed a dynamic smart key management system. DSKMS included the processes of key generation and distribution, key revocation and rekeying, adding new nodes to the network and node movement. In DSKMS, we made use of four types of keys namely private key, cluster key, pairwise key and path key and in the presented paper we explained how these keys are generated and distributed. In order to improve the functioning of DSKMS, we also used the fuzzy systems of FSDS1 and FSDS2 in the path key generation processes and in adding new node to the network. Ultimately, we compared the performance of DSKMS with that of the three other key management schemes of HKMS [19], LEAP+ [20], and EAHKM [21] in terms of communication overload, energy consumption, the required memory space and the effect of the cryptanalyzed nodes on the network. The results of simulation demonstrate the superiority of DSKMS to other key management schemes. DSKMS consumes less energy and elongates the network lifetime and enhances the resilience and resistance of the network to cryptanalysis attacks.

References

1. Qiu, T., Chen, N., Li, K., Qiao, D., & Fu, Z. (2017). Heterogeneous ad hoc networks: Architectures, advances and challenges. *Ad Hoc Networks*, *55*, 143–152.
2. Alaba, F. A., Othman, M., Hashem, I. A. T., & Alotaibi, F. (2017). Internet of things security: A survey. *Journal of Network and Computer Applications*, *88*, 10–28.
3. Afsar, M. M., Tayarani, N., & Mohammad, H. (2014). Clustering in sensor networks: A literature survey. *Journal of Network and Computer Applications*, *46*, 198–226.
4. Jawad, H., Nordin, R., Gharghan, S., Jawad, A., & Ismail, M. (2017). Energy-efficient wireless sensor networks for precision agriculture: A review. *Sensors*, *17*(8), 1781.
5. Annapurna, H. S., & Siddappa, M. (2015). A technique for multi-tier key distribution for securing group communication in WSN, emerging research in computing, information. *Communication and Applications*, 273–279.
6. Bhushan, B., & Sahoo, G. (2018). Recent advances in attacks, technical challenges, vulnerabilities and their countermeasures in wireless sensor networks. *Wireless Personal Communications*, *98*(2), 2037–2077.
7. Yang, K. (2014). *Wireless sensor networks, principles, design and applications* (pp. 187–215). London: Springer.
8. Chen, L. (2013). *Wireless network security* (pp. 129–221). Beijing: Springer.
9. Yang, Y., Wu, L., Yin, G., Li, L., & Zhao, H. (2017). A survey on security and privacy issues in internet-of-things. *IEEE Internet of Things Journal*, *4*(5), 1250–1258.
10. El-Bendary, M. (2015). *AM and others, developing security tools of WSN and WBAN networks applications* (pp. 79–95). Tokyo: Springer.
11. Yang, G. C., Ao, S. L. G., & An, L. (2015). *IAENG transactions on engineering technologies* (pp. 615–628). London: Springer.
12. Wang, Y., & Zhao, J. (2017). Key management scheme for wireless sensor networks. In *International wireless internet conference* (pp. 272–283). Cham: Springer.
13. Huang, J.-M., Yang, S.-B., & Dai, C.-L. (2013). An efficient key management scheme for data-centric storage wireless sensor networks. *IERI Procedia*, *4*, 25–31.
14. Yousefpoor, M. S., & Barati, H. (2019). Dynamic key management algorithms in wireless sensor networks: A survey. *Computer Communications*, *134*, 52–69.
15. Kodali, R. K. & Chougule, S. (2013). Hybrid key management technique for WSNs. In *International conference on heterogeneous networking for quality, reliability, security and robustness* (pp. 854–865).
16. Thevar, G. K. C., & Rohini, G. (2017). Energy efficient geographical key management scheme for authentication in mobile wireless sensor networks. *Wireless Networks*, *23*(5), 1479–1489.
17. Bekara, C., Laurent-Maknavicius, M. (2009). *Wireless and mobile network security* (pp. 613–648). France: Wiley.
18. Hassanien, A. E., Kim, T. H., Kacprzyk, J., & Awad, A. (2014). *Bio-inspiring cyber security and cloud services: Trends and innovations*. New York: Springer.
19. Zhang, Y., Li, X., Liu, J., Yang, J., & Cui, B. (2012). A secure hierarchical key management scheme in wireless sensor network. *International Journal of Distributed Sensor Networks*, *8*(9), 547471.
20. Zhu, S., Setia, S., & Jajodia, S. (2006). LEAP+: Efficient security mechanisms for large-scale distributed sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, *2*(4), 500–528.
21. Messai, M. L., Seba, H., & Aliouat, M. (2015). A new hierarchical key management scheme for secure clustering in wireless sensor networks. In *International conference on wired/wireless internet communication* (pp. 411–424).
22. Ahlawat, P., & Dave, M. (2018). An attack model based highly secure key management scheme for wireless sensor networks. *Procedia Computer Science*, *125*, 201–207.
23. Aissani, S., Omar, M., Tari, A., & Bouakkaz, F. (2018). μKMS: Micro key management system for WSNs. *IET Wireless Sensor Systems*, *8*(2), 87–97.
24. Seo, S.-H., Won, J., Sultana, S., & Bertino, E. (2015). Effective key management in dynamic wireless sensor networks. *IEEE Transactions on Information Forensics and Security*, *10*(2), 371–383.

25. Blom, R. (1984). An optimal class of symmetric key generation systems. In *Workshop on the theory and application of of cryptographic techniques* (Vol. 209, pp. 335–338).
26. Blundo, C., De Santis, A., Herzberg, A., Kutten, S., Vaccaro, U., & Yung, M. (1992). Perfectly-secure key distribution for dynamic conferences. *Annual international cryptology conference* (pp. 471–486).
27. Rahman, M., & Sampalli, S. (2015). An efficient pairwise and group key management protocol for wireless sensor network. *Wireless Personal Communications*, *84*(3), 2035–2053.
28. Erfani, S. H., Javadi, H. H. S., & Rahmani, A. M. (2015). A dynamic key management scheme for dynamic wireless sensor networks. *Security and Communication Networks*, *8*(6), 1040–1049.
29. Hosen, A. S. M. S., & Cho, G. H. (2014). A robust key management scheme based on node hierarchy for wireless sensor networks. In *International conference on computational science and its applications* (pp. 315–329).
30. Siddique, N., & Adeli, H. (2013). *Computational intelligence: Synergies of fuzzy logic, neural networks and evolutionary computing*. London: Wiley.
31. De Silva, C. W. (2018). *Intelligent control: Fuzzy logic applications*. Boca Raton: CRC Press.
32. Cayirci, E., & Rong, C. (2009). *Security in wireless ad hoc and sensor networks* (pp. 121–141). Norwey: Wiley.
33. Rathore, H. (2016). *Case study: A review of security challenges, attacks and trust and reputation models in wireless sensor networks, mapping biological systems to network systems* (pp. 117–175). Zurich: Springer.
34. Delfs, H., Knebl, H., & Knebl, H. (2015). *Introduction to cryptography* (pp. 1–483). Berlin: Springer.
35. Boyle, D. E., & Newe, T. (2009). On the implementation and evaluation of an elliptic curve based cryptosystem for Java enabled wireless sensor networks. *Sensors and Actuators A: Physical*, *156*(2), 394–405.
36. Sahingoz, O. K. (2013). Large scale wireless sensor networks with multi-level dynamic key management scheme. *Journal of Systems Architecture*, *59*(9), 801–807.
37. Oreku, G. S., & Pazynyuk, T. (2016). *Security in wireless sensor networks* (pp. 1–87). Zurich: Springer.
38. He, X., Niedermeier, M., & De Meer, H. (2013). Dynamic key management in wireless sensor networks: A survey. *Journal of Network and Computer Applications*, *36*(2), 611–622.
39. Ferng, H.-W., Nurhakim, J., & Horng, S.-J. (2014). Key management protocol with end-to-end data security and key revocation for a multi-BS wireless sensor network. *Wireless Networks*, *20*(4), 625–637.
40. Azzabi, T., Farhat, H., & Sahli, N. (2017). A survey on wireless sensor networks security issues and military specificities, In *International conference on IEEE, advanced systems and electric technologies* (IC_ASET) (pp. 66–72).
41. Rezai, A., Keshavarzi, P., & Moravej, Z. (2017). Key management issue in SCADA networks: A review. *Engineering Science and Technology, An International Journal*, *20*(1), 354–363.
42. Yaqoob, I., Ahmed, E., Hashem, I. A. T., Ahmed, A. I. A., Gani, A., Imran, M., et al. (2017). Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges. *IEEE Wireless Communications*, *24*(3), 10–16.

Mohammad Sadegh Yousefpoor received B.Sc. degree in computer science from Dezful Branch, Payame Noor Universtiy, Dezful, Iran, in 2010. He is currently working toward the M.Sc. degree in Artificial Intelligence Computer Engineering at Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran. His research interests include Wireless sensor networks (WSNs), internet of things (IoT), Machine learning, Pattern recognition, Cryptography and Network security.

Hamid Barati received his B.S. degree in Computer Hardware Engineering, M.S. degree in Computer Systems Architecture Engineering and Ph.D. degree in Computer Systems Architecture Engineering in 2005, 2007 and 2015 respectively. Currently he is faculty of Islamic Azad University, Dezful Branch, Iran. His major research experiences and interests include Mobile Ad-Hoc Networks, Interconnection Networks & Energy-Efficient Routing and Security issues in Wireless Sensor Networks.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.