Journal of Information Assurance and Security.
ISSN 1554-1010 Volume 7 (2012) pp. 324-337

A Dual-Sink Secure Routing Protocol for Wireless Sensor Networks

Aly M. El-Semary[1,2] and Mohamed M. A. Azim[1,3]

[1] College of Computer Science and Eng., Taibah University
Al-Madina Al-Munawara, Saudi Arabia
{aelsemary, mzayed}@taibahu.edu.sa

[2] Systems and Computer Engineering Dept., Faculty of Engineering,
Al-Azhar University, Cairo, Egypt
alyelsemary@azhar.edu.eg

[3] Faculty of industrial Education, Beni-Suef University,
Beni-Suef, Egypt.
mmazim@ieee.org

***Abstract*: The recent advancements in electronics and communications technology have enabled the extensive deployment of wireless sensor networks (WSNs) into a variety of areas including homeland security, military systems, industrial, and health care. WSNs are typically deployed in unattended, hostile and harsh environment which makes them subject to several attacks such as sinkhole, spoofing or Sybil, wormhole, and selective forwarding. Therefore, developing secure, load-balanced and scalable routing protocols for WSNs has become imperative. In the recent literature, most of the routing protocols take the routing decision based on information collected from neighboring nodes. However, this allows a malicious sensor node to deceive its neighbors and forward their packets through it. This makes the network's nodes vulnerable to various attacks. Therefore, in this paper we propose a Dual Sink Secure Routing Protocol (DSSRP) for WSNs. The DSSRP is a scalable protocol for secure routing in which, the routing decision of a node is based on its own information. Thus it cannot be deceived by any other sensor node. Extensive simulation results indicate clearly that, the DSSRP protocol achieves very high confidentiality, integrity, authentication, and non-repetition with a very low overhead. In addition, the proposed protocol exhibits a grateful performance under attack-free conditions by evenly distributing the network load among the deployed nodes thus extending the network lifetime.**

***Keywords*:** secure routing, WSN, dual sink, sinkhole attack; selective forwarding.

I. INTRODUCTION

A Wireless Sensor Network (WSN) is a specific class of wireless Ad-Hoc networks in which hundreds or thousands of sensor nodes are collaborating together to accurately measure a physical phenomenon from the environment or to monitor a remote site. The WSN has a flexible architecture and consists of large number of small sensor nodes that are distributed around a remote site. The sensor nodes are able to sense physical phenomena and report the sensed data to other nodes around the area. The base station is responsible for collecting all the sensors data. Then the collected data is imported into a database and can be visualized.

Due to the recent advances in electronics and wireless communication technologies, wireless sensor networks (WSNs) have been deployed into variety of areas including homeland security, military systems, agriculture, and health care. In such networks, typical sensor nodes are small in size with limited communication and storage, computing capabilities, and are powered by batteries. These small sensor nodes are subject to many kinds of attacks such as sinkhole, select forwarding, spoofing, and wormhole. This requires the development of secure and energy-efficient routing protocols to protect the network against such attacks. However, due to the limited capabilities of sensor nodes, providing security and privacy to a sensor network is a challenging task.

Several recent contributions to the literature have addressed the routing issues in sensor networks [1–12]. Some of these protocols [1–5] concentrate on energy utilization of the deployed sensors. Surveys can be referred to in [6, 7]. Other protocols are designed to provide a security against specific types of attacks such as [8–10]. A survey can be found in [11].

The authors in [11] pointed out that, the challenge of designing security protocols for sensor networks lies in establishing a secure communication infrastructure, before any routing fabric has been established. Hence, sensor networks employing protocols that forward packets based on information collected from other sensor nodes may be deceived by a malicious node. This fake node will mislead the forwarding nodes by forwarding their packets through it and then enable several types of WSNs attacks. Therefore, we presented in [12] a two-tier Energy Efficient Secure Routing Protocol (EESRP) for WSNs in which the forwarding criterion is based only on the local data of the node itself. Thus the forwarding node cannot be deceived by any other node. However, the proposed protocol works only in WSNs with single sink node. In this case, the nodes near the sink are more likely to use up their energy because they have to forward all the data generated by the nodes farther away [13].

In this paper, we extend our work in [12] to more general scenario in which a sensor network has dual sink nodes. The proposed protocol improves the network security, load balancing as well as enhances the protocol scalability.

The remainder of this paper is organized as follows. Section II gives an overview of our related work. Section III presents the proposed DSSRP protocol. The security aspects of the proposed protocol are discussed in section IV. The experimental results are discussed in Section V. Conclusion is presented in Section VI.

II. Related Work

Wireless sensor networks (WSN) are typically deployed in an unattended environment, which makes them vulnerable to variety of attacks. These attacks include selective forwarding, sinkhole, and spoofing attacks. In selective forwarding attack, malicious nodes may not forward specific messages. A simple aspect of this attack is when a malicious node acts like a black hole by refusing to forward every observable packet. The sinkhole attack prevents the base station from obtaining complete and correct sensing data, thus forming a serious threat to higher-layer applications. It is achieved by making a compromised node look attractive to its neighbor nodes with respect to the routing metrics. Consequently, the attacker manages to draw as much traffic as possible that is designated to the base station. By involving itself in the routing process, it is then able to launch more sever attacks such as selective forwarding, modifying or dropping the received packets. In a spoofing attack, an attacker can easily inject fake packets by impersonating another sender.

The proposed DSSRP protocol implements the security by using the *Advanced Encryption Standard* (*AES*) algorithm [14], *Message-Digest* (*MD5*) algorithm [17], and *μTESLA protocol* [18] as symmetric broadcast authentication protocol. The AES algorithm [14] is a symmetric-key block cipher algorithm has a fixed block size of 128-bits and a key size of 128, 192, or 256-bits. AES has become standardized encryption algorithm and the default choice in numerous applications, including the standard WSN technologies IEEE 802.15.4 [15] and ZigBee [16]. The MD5 [17] algorithm takes as input a message of arbitrary length and produces an output of a 128-bit "fingerprint" or "message digest" of the input. Therefore, it is usually used to verify data integrity. The μTESLA [18] has been proposed for broadcast authentication in distributed sensor networks. Generally the broadcast authentication is implemented using asymmetric mechanisms but due to the high communication, computation, and storage overheads of the asymmetric cryptographic mechanisms, it is impractical to implement them in resource constrained sensor networks. Therefore, μTESLA introduced asymmetry by delaying the disclosure of symmetric keys. In this protocol, the network life time is divided into n time intervals and chain of authentication keys $K = \{k_0, k_1, \ldots, k_n\}$ is generated. The keys in K is linked to each other by a one-way function and they are obtained by first choosing a random value k_n as the last key in K and then continuously executing a one-way function f to compute all the other keys: $k_i = f(k_i+1)$, $0 \le i \le n-1$. Each k_i is assigned to authenticate all broadcast messages sent in the i^{th} time interval, $1 \le i \le n$, and k_0 is the initial key which refers to the commitment of K. If a key k_j is given, only the previous keys k_i can be computed using f, $0 \le i \le j-1$, but the later keys k_i cannot be computed, $j+1 \le i \le n$. Therefore, with the knowledge of k_0, any other key in K can be authenticated by just performing f.

III. Proposed Protocol

In this section we present our proposed Dual-Sink Secure Routing Protocol (DSSRP) for wireless sensor networks. The proposed DSSRP protocol enhances the EESRP [12] protocol by (1) increasing key length from 56 bits to 128 bits, (2) changing the shared key at regular intervals or when it is compromised (3) enhancing the protocol scalability by extending the protocol to deal with dual sink nodes instead of a single sink node. Moreover, this dual sink architecture can be cascaded for large-scale WSN. The DSSRP is constructed from two sub protocols: Next Node Selection Protocol (NNSP) and Network Protection Protocol (NPP). The NNSP is responsible for routing data packets while the NPP protects them during their traveling from a source node to either $sink_1$ or $sink_2$. The rest of this section presents the protocol notations, the network model, the NNSP protocol, and the NPP protocol, in Section *A*, *B*, *C*, and *D*, respectively.

A. DSSRP's Notations

The notations used throughout the DSSRP are introduced in this section. All encryptions and decryptions are achieved by using the advanced encryption standards (AES) algorithm.

- $E(m, k)$: the encryption of message m with the key k.
- $D(C, k)$: the decryption of the cipher text C with the key k.
- $E(m, k, IV)$: the encryption of a message m with the key k and the initial vector IV.
- $D(C, k, IV)$: the decryption of C with the key k and the initial vector IV.
- *MD5* (m): gets the hash value of m using the MD5 algorithm.
- $x \| y$: the concatenation of x and y.

B. Network Model

The network model used throughout this paper is depicted in Figure1. It is composed of a set of randomly distributed wireless sensor nodes, two sink nodes denoted by $sink_1$ and $sink_2$, and two workstations referred as BS_1 and BS_2. Each sink node is connected directly to a workstation (e.g., $sink_1$ is connected to $sink_2$) and the two workstations are connected together through the Internet. The sink nodes can be positioned anywhere on the network's border but it is better to put them at the border around a furthest two corners in network field. The underlying network model is similar to the one used in paper [12] but the underlying network model is employing two sink nodes that greatly enhances the scalability by cascading the proposed network architecture. The Finally, It is required that every authorized deployed sensor in the network field has a unique identification number (ID) and follows the protocol steps.

C. NNSP Protocol

The Next Node Selection Protocol (NNSP) is responsible for forwarding data packets to a next hop toward either of the two deployed sink nodes. This protocol extends our proposed RWRP protocol in [12]. Similar to RWRP, the significance of the NNSP protocol is that the routing decision of a sensor

node is not affected by its neighbors. However, the NNSP surpasses the RWRP in the distribution of the network load evenly among the deployed wireless sensor nodes. Therefore, it prolongs the network life-time.

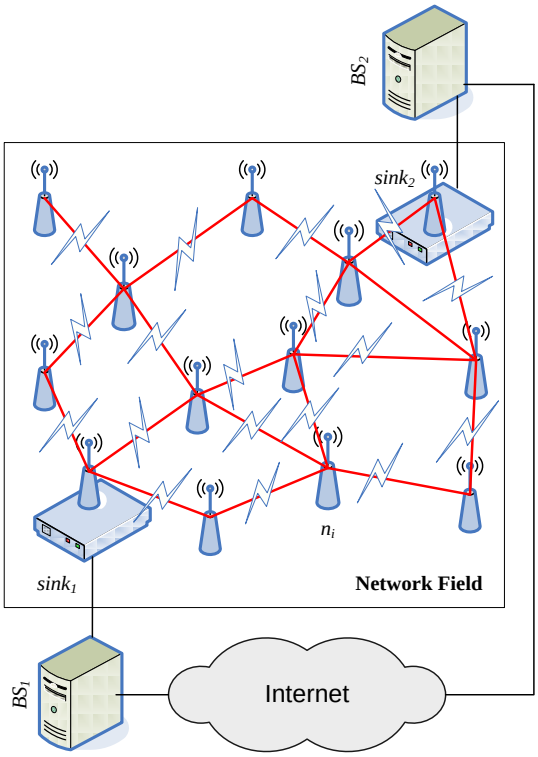


Figure 1. Network Field.

Unlike the RWRP protocol which selects only the forwarding node, the NNSP protocol first, selects a sink node toward which the data packets will be forwarded. Second, it selects forwarding nodes along the path towards the predetermined sink node. This is achieved through the selection probabilities of the sink nodes and forwarding nodes described in Equation 1 and 2, respectively.

Where sp_{sink} $(ns_{i,j})$ is the selection probability of the source node n_i to forward a data packet into the sink node $sink_j$, $i = 1, 2, \ldots, M$, where M refers to the number of sensor nodes in the network field, $j = 1, 2$ and $hs_{i,j}$ is the number of hops from a sensor node n_i to the sink node $sink_j$. The selection probability $sp_{sink}(ns_{i,j})$ is designed such that it gives the highest probability to $sink_j$ that has the smallest number of hops from the node n_i.

Where sp_{fn} $(f_{i,j})$ denotes the selection probability of choosing a node n_j from the routing table of the forwarding node n_i during the forwarding process, $pk_{i,j}$ is the number of sent packets from the node n_i to the node n_j and N is the number of parents of the node n_i.

Note that, the selection probability sp_{fn} $(f_{i,j})$ is constructed such that it gives the highest probability to the parent n_j associated with the minimum number of $pk_{i,j}$. The node n_i will use the sibling instead of parent nodes in case of there is no parents exist for the node n_i.

During the selection process, the NNSP protocol differentiates between two types of nodes: a sensing node and a forwarding node. A sensing node is the source node that senses the environment. The forwarding node is a node along the path from the sensing node to a sink node. The sensing node is responsible for choosing a sink node and a forwarding node towards the selected sink node. The forwarding node is in charge of choosing only a next forwarding node.

In the NNSP protocol, each sensing node should first determine the sink node toward which a packet will be forwarded. Here, a sink node is selected by first calculating the selection probabilities sp_{sink1} and sp_{sink2} of the first and second sink nodes based on Equation 1, respectively. Next, a random number r is generated. If $r \leq sp_{sink1}$, the data packet will be forwarded to the $sink_1$. Otherwise, the data will be forwarded to $sink_2$ as described by steps 2 to 10 in the algorithm shown in Figure 2.

```
NNSP_SelectionAlgorithm (ni )
1     sink ← null
2     IF ni is a sensing node
3           spsink1 ← evaluation of Equation 1 for the 1st sink
4           spsink2 ← evaluation of Equation 1 for the 2nd sink
5           r ← generate a random number
6           IF r ≤ spsink1
7                 sink ← 1st sink
8           ELSE
9                 sink ← 2st sink
10          END
11    ELSE  // ni is a forwarding node
12          sink ← extract the sink from packet received by ni
13    END
14    M ← no. of parents of ni // siblings are used if no parents exist
15    spfn ← { }
12    FOR j =1 to M
13          spfn (fi,j)  ← evaluation of Equation 1 for nj
14          spfn ← spfn U spfn (fi,j)
15    END
16    construct a probability line by adding up all spfn (fi,j)
17    r ← generate a random number
18    select nj that is matched with r
END
```

Figure 2. NNSP Selection Algorithm.

For determining a next forwarding node to either a sensing or forwarding node n_i, the selection probabilities of all parents (or sibling in case there is no parent nodes for n_i) of the node n_i are calculated based on Equation 2.Next, a probability line is constructed by summing up all the selection probabilities. Then, a random number r is generated. Finally, the node n_j that the value of r lies within its selection probability range on the probability line will be chosen as the next forwarding node. This is described by steps 11 to 18 in Figure 2.

To demonstrate how the NNSP protocol works, we present the following example in which a sample network is constructed as shown in Figure 3. The network consists of eleven nodes and two sink nodes. A bidirectional wireless

communication link between sensor nodes within the transmission range of each other is represented by a double-headed arrow in the manner shown in Figure 3. The notations used in this figure are as follows. Each node is represented by two ellipses and two triangles as well as the node ID in the center. The ellipses contain the numbers of sent packets to each parent of the node toward $sink_1$ or $sink_2$ direction based on a solid or dashed ellipse, respectively. For example, n_6 has two parent nodes (n_4 and n_5) toward $sink_1$ while it has three parent nodes (n_7, n_8, and n_9) towards $sink_2$. According to Figure 3, n_6 has sent 20 and 18 packets to n_4 and n_5, respectively. Also, the solid and the dashed triangles of a node represent the number of hopes from the node to $sink_1$ and $sink_2$, respectively. For example, n_6 is 3 hops apart from $sink_1$ and 5 hops away from $sink_2$.

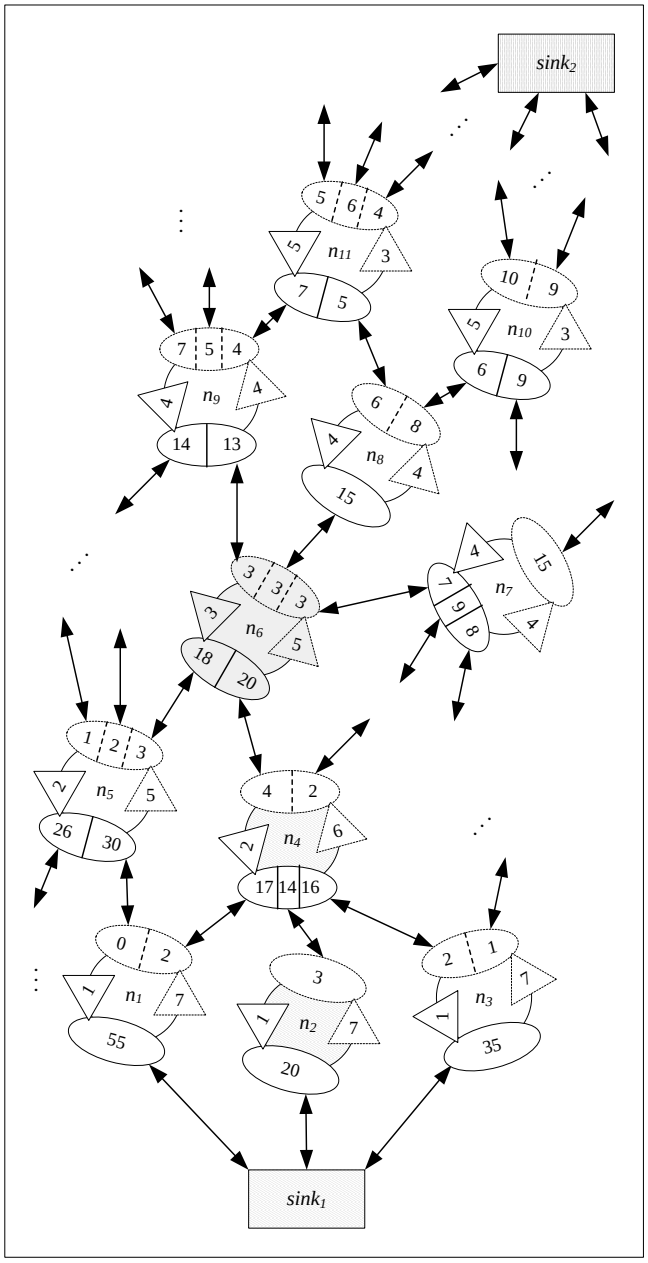


Figure 3. A sample network example.

Now, let's consider that n_6 is the sensing node in the network shown in Figure 3 and we need to find out which path it might take and which sink node is chosen. Based on the proposed NNSP algorithm, n_6 calculates selection probabilities for both sink nodes from Equation 1. Then we have the selection probabilities of 0.625 and 0.375 for $sink_1$ and $sink_2$, respectively. Next, n_6 will construct the probability line by summing up the selection probabilities. Next, n_6 will generate a random number r which is assumed to be 0.3 as shown in Figure 4a. Finally, as the value of r lies within the selection probability range of $sink_1$ on the probability line, $sink_1$ is selected to be the final destination for the packet initiated by n_6.

After selecting $sink_1$ by n_6, it has to select one of its parent nodes (n_4 and n_5) as the next forwarding node. To achieve this task, n_6 calculates the selection probabilities of n_4 and n_5 based on Eq. 2 which are equal to 0.474 and 0.526, respectively. n_6 will construct the probability line by summing up the selection probabilities of n_4 and n_5. Next, n_6 will generate a random number r which is assumed to be 0.25 as shown in Figure 4b. Finally, n_4 is selected as the next forwarding node as the value of r lies within its selection probability range on the probability line as shown in Figure 4b.

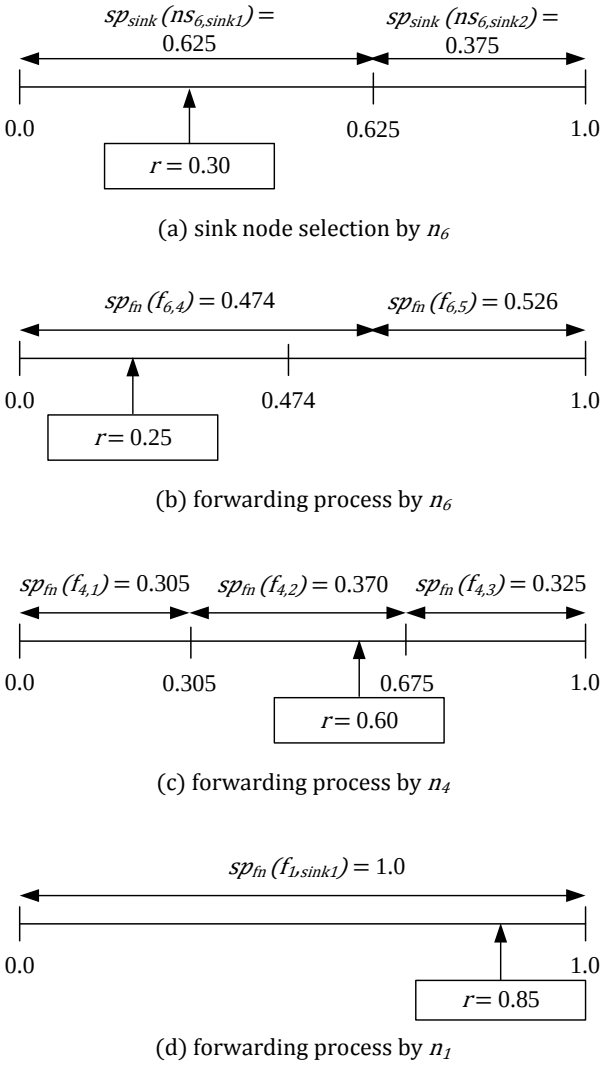


Figure 4. The Forwarding process from n_6 to $sink_1$.

When the data packet reaches n_4, it starts selecting one of its parent nodes n_1, n_2 and n_3 which have selection probabilities of 0.305, 0.370, and 0.325 based on Equation 2 respectively. Similarly, n_4 will construct the probability line by summing up the selection probabilities of n_1, n_2 and n_3. Then, n_4 will

generate a random number r which is assumed to be 0.6 which corresponds to the selection probability of n_2 on the probability line as shown in Figure 4c.

Finally in the same way, n_2 will forward the data packet to $sink_1$. Hence, the overall forwarding path is $n_6 \rightarrow n_4 \rightarrow n_2 \rightarrow sink_1$ as indicated in Figure 3 by the hashed sensor nodes.

D. Network Protection Protocol

The Network Protection Protocol (NPP) is an enhanced version of our SRP protocol proposed in [12]. In terms of security, the NPP protocol improves the network security by (1) replacing the DES encryption algorithm by the AES, hence, increasing the key length to 128 bits instead of 56 bits and (2) providing a mechanism to change the shared key K_s regularly throughout the network lifetime instead of using the fixed key.

Similar to the SRP protocol, NPP protocol achieves confidentiality, integrity, and authentication through four phases of operations: (1) initialization, (2) Configuration, (3) Elimination, and (4) Secure forwarding. The NPP protocol implements the security by using the Advanced Encryption Standard (AES) algorithm [14], Message-Digest (MD5) algorithm [17], and µTESLA protocol [18] as symmetric broadcast authentication protocol. Furthermore, a time stamp is used to prevent repetition of packets. The above mentioned security services will be described throughout the protocol phases.

1) Initialization Phase

The initialization phase is concerned with nodes before their actual deployment into the network field. This initialization phase is described by the algorithm shown in Fig. 5. First, the set of key chain $K = \{k_0, k_1, \ldots, k_n\}$ used in the µTesla is generated by randomly selecting k_n and the rest of keys k_i are generated as $k_i = MD5\ (k_{i+1})$ where $0 \leq i \leq n\text{-}1$.

Next, each sensor node n_i must be initialized with five secret keys, two initial vectors, challenge value (ch_i), and a message authentication code (MAC_i).

The five secret keys are denoted by k_0, k_s, k_m, $k_{i,1}$, and $k_{i,2}$, each of length 128 bits. k_0 is the commitment key of the key chain K. The k_s is a secret key shared only among all authorized nodes deployed into the network field. k_m is the master key used to regularly change k_s. The keys $k_{i,1}$ and $k_{i,2}$ are secret keys shared only between the node n_i and the $sink_1$ and $sink_2$, respectively. Also the node n_i should be initialized with two initial vectors IV_s and IV_i, each of length 128 bits. The challenge value is a small random number to be used in the challenge response protocol during the configuration phase. Finally, the node n_i is also initialized with a MAC_i which is used to detect and remove unauthorized nodes during the elimination phase. The generation of MAC_i is shown in Fig. 5.

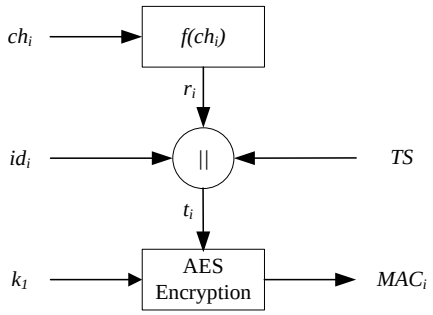


Figure 5. Message authentication code generation for node *i*.

The MAC_i is an encrypted version of the token t_i using the AES algorithm with the key k_1 that belongs to the key chain K. The token t_i is the concatenation of id_i , r_i, and TS, where id_i is a unique identification number for the node i, r_i is the response of a *challenge*-response function, $f_{cr}(ch_i)$. The $f_{cr}()$ is a function known by all authorized nodes deployed in the network field. Finally, TS is a unique time stamp generated by $sink_1$ to assure the refreshment of the MAC_i.

2) Configuration Phase

The configuration phase is responsible for building the routing table of each authorized node in the network field. In our NPP protocol, nodes in the routing table toward a certain sink node are classified into four categories: *1) parent node, 2) sibling node, 3) child node and 4) combination*. A parent node is a node in the transmission range of another sending node and having a hop count one less than the sending node. A sibling node is a node in the transmission range of another sending node and having the same hop count as the sending node. A child node is a node in the transmission range of another sending node and having a hop count one more than the sending node. A *combination* means a node can be for example a parent node toward $sink_1$ while being a child node towards $sink_2$. The routing table establishment of each node is achieved through two rounds. The first round is initiated by $sink_1$. After the completion of the first round, $sink_1$ notifies $sink_2$ to start the second round.

During the first round of the configuration phase, the setup packet created by a node n_i is constructed as "*setS1*.MAC_i.$C1_i$," where the keyword *setS1* is used to denote that this is a setup packet towards $sink_1$, MAC_i is the message authentication code generated during the initialization phase for the node i, and $C1_i$ is the encryption of the message $M1_i$ with the shared key k_s using the AES (i.e., $C1_i = E(M1_i, k_s)$). The message $M1_i$ is the concatenation of id_i, ch_i, x_i, y_i, $h1_i$, and $TS1_i$, where id_i and ch_i are the identification number and challenge value of the node n_i, respectively. The parameters x_i and y_i denote the coordinates of the node n_i in x and y-directions, respectively. The parameter $h1_i$ refers to the number of hops between the $sink_1$ and the node n_i. Finally, $TS1_i$ is a unique time stamp generated by the node n_i. This routing process is demonstrated step-by-step via the network example in Figure 6.

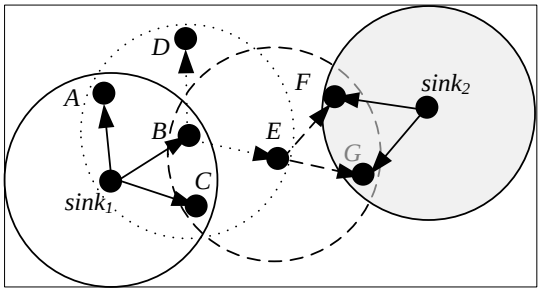


Figure 6. Example of building routing tables.

The first configuration round is started when $sink_1$ creates its setup packet with hop count of zero (i.e., $h1_{sink1} = 0$) and broadcasts it to all sensor nodes within its transmission range. In this case, the nodes *A*, *B*, and *C* in the solid circule will receive the setup packet. Since the hop counts of these nodes have not set before, each of the nodes *A*, *B*, and *C* marks the $sink_1$ as its parent and sets its hop-count associated with $sink_1$ the value of 1.

Next, every node with hop-count of l apart from $sink_1$, in this case *A*, *B*, and *C*, constructs its own setup packet and broadcasts it to all nodes within its transmission range. For example, suppose that the node *B* in the doted circle sends its setup packet which in turn will be received by its neighbor nodes: $sink_1$, *A*, *C*, *D*, and *E*. Each of these nodes decrypts the setup packet with the k_s to extract the included parameters and accordingly updates its routing table. The $sink_1$ marks the node *B* as a child node since the value of $h1_{sink1}$ is less than the value of $h1_B$. The nodes *A* and *C* marks node *B* as a sibling node because each of the values of $h1_A$ and $h1_C$ is equal to the value of $h1_B$. Finally, each of the nodes *D* and *E* sets its hop count to 2 (i.e., $h1_B$ +1) and marks the node *B* as its parent since neither the hop count $h1_D$ nor the hop count $h1_E$ is set before. Thus, each of these nodes creates its setup packet and broadcasts it to all nodes in its transmission range. Therefore, according to the setup packet sent from the node *E* as an example, each of the nodes *F* and *G* sets its hop count to the value of 3 (i.e., $h1_E$ +1) and marks the node *E* as a parent. Then the nodes *F* and *G* continue in the same way. After finishing the first round of the configuration, *sink1* notifies *sink2* to start the second configuration round.

During the second round, a new setup packet format is considered as "setS2. $C2_i$." Where the keyword *setS2* refers to the second round initiated by $sink_2$ while $C2_i$ is the encryption of the message $M2_i$ with the shared key k_s using the AES (*i.e.*, $C2_i = E\ (M2_i, k_s)$). The message $M2_i$ is the concatenation of id_i, ch_i, $h2_i$, and $TS2_i$, where id_i and ch_i are as before the identification number and the challenge value of the node *i*, respectively. The parameter $h2_i$ refers to the number of hops between the $sink_2$ and the node *i*. Finally, $TS2_i$ is a unique time stamp generated by the node *i*.

After $sink_2$ is notified by $sink_1$, it creates its setup packet with hop count of zero (i.e., $h2_{sink2} = 0$) and broadcasts it to all sensor nodes within its transmission range. In our example, the nodes *F* and *G* in the circle filled with gray color will receive the setup packet. Each of these nodes will decrypt the setup packet with the k_s to extract the included parameters and accordingly updates its routing table. Since the hop counts ($h2_F$ and $h2_G$) of these nodes have not set before, each of the nodes *F* and *G* marks the $sink_2$ as its parent and sets its hop-count associated with $sink_2$ the value of 1.

Next, every node with hop-count of l apart from $sink_2$, in this case, each of the nodes *F* and *G* constructs its own setup packet and broadcasts it to all nodes within its transmission range. For example, suppose that node *F* sends its setup packet which in turn will be received by its neighbor nodes: $sink_2$, *E*, and G. Each of these nodes also decrypts the setup packet with the k_s to extract the included parameters and accordingly updates its routing table. The $sink_2$ marks the node *F* as a child node since the value of $h2_{sink1}$ is less than the value of $h2_F$. The node *G* marks node *F* as a sibling node because the values of $h2_F$ and $h2_G$ are equal. Finally, the node *E* sets its hop count to 2 (i.e., $h2_F$ +1) and marks the node *F* as its parent. Note that the node *E* marks the node *F* as a child during the first round and a parent during the second round towards $sink_1$ and $sink_2$, respectively. Therefore, the node *E* also marks the node *F* as a *combination*. This process is executed hop-by-hop as in the first round until finishing the second round of the configuration phase. When the second round is completed, $sink_2$ notifies $sink_1$ to start the elimination phase.

Note that, each authorized node n_i in the network field also has a field called *active* to indicate whether the node n_i is operational or not. Operational means that the node n_i can receive or forward data packets. The *active* field is set to true if the node n_i is operational. Otherwise, it is set to false. Up to this phase, all deployed nodes are not operational until the elimination phase is executed.

3) Elimination phase

This phase is used to eliminate any malicious nodes that might be injected into the routing tables. It is started when $sink_1$ broadcasts an elimination packet which has the form "*EP.AM*," where *EP* is a keyword to denote that this is an elimination packet and *AM* is an authentication message. *AM* is the encryption of a message *m* using the AES algorithm with the shared key k_s and the initial vector IV_s (*i.e.*, $AM = E\ (m, k_s, IV_s)$). The message *m* is formed from the concatenation of id_{sink1}, k_1, and TS_{sink1}. Where id_{sink1} is the identification number of $sink_1$, k_1 belongs to the key chain *K* generated during the initialization phase, and TS_{sink1} is a new time stamp.

When an authorized node n_i receives the elimination packet, it first obtains the message *m* by decrypting the authentication message *AM (i.e.*, $m = D\ (AM, k_s, IV_s)$. Next, it extracts the values of id_{sink1}, k_1, and TS_{sink1} from *m*. Then, the node examines the time stamp TS_{sink1} to check whether the received elimination packet is a reply packet or not. If so, it simply ignores the packet. Otherwise, it authenticate k_1 by performing k = MD5 (k_1). If k is not equal to k_0 which is stored during the initialization phase, the node n_i simply ignores the packet. Otherwise, the node n_i applies the following process for each node n_j in its routing table. It decrypts MAC_j associated with each node n_j to reveal the token t_j (i.e., $t_j = D\ (MAC_j, k_s, IV_s)$), where t_j was constructed from the concatenation of id_j, r_j, and TS_j during the configuration phase as shown in Figure 5. These values are compared with the data received from the node *j* during the configuration phase. The received data includes the challenge ch_j and time stamp TS_j of the node n_j. If the response r_j is matched with the challenge ch_j and the two time stamps are matched, the node n_j is considered as an authorized node. Thus, the node n_i should put the node n_j into an operational state. This is achieved by setting the *active* field of the record corresponding to the node n_j in the routing table of the node n_i to true. Otherwise, the

node n_i will consider the node n_j as a malicious node. Consequently, the node n_j will be removed from the routing table of the node n_i.

4) Secure Forwarding Phase

This secure forwarding phase is concerned with routing data packets in a secure way. A data packet is intended to be delivered through a determined path from a sensing node to a sink node. The data packet is processed by all nodes along the path during its transmission. The nodes along the route from the source to the sink node can be classified based on their functionality into three categories: 1) sensing node, 2) forwarding node, and 3) sink node. The operation achieved by each category is presented in the rest of this section.

When a node n_i belonging to the first category has a sensed data d_i, it has first to choose the sink node $sink_q$ to which d_i will be sent, where q =1,2. Note that, the $sink_q$ is chosen by the node n_i through applying the NNSP protocol described in Section C. Next, the node n_i will apply confidentiality, integrity and non-repetition for d_i between the node n_i and the $sink_q$ as described in Figure 7.

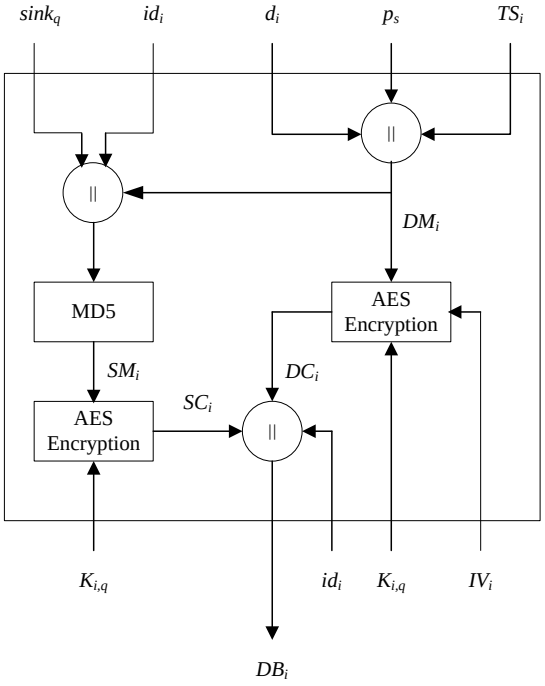


Figure 7. Applying confidentiality and integrity for the sensed data d_i by a node n_i.

Figure 7 shows that the confidentiality is achieved through constructing DC_i which is encrypted version of the message DM_i using the AES algorithm. The AES algorithm is used in the cipher Block Chain (CBC) mode of operation with the key $k_{i,q}$ and the initial vector IV_i (i.e. $DC_i = E(DM_i , k_{i,q} , IV_i)$). The $k_{i,q}$ is the key shared between node n_i and the $sink_q$. The message DM_i is the concatenation of the three parameters namely; the sensed data d_i, the generated packet sequence p_s, and the generated time stamp TS_i. At the same time, the integrity is applied through constructing the signature SC_i by encrypting the signature message SM_i using the AES algorithm with the key $k_{i,q}$ (i.e. $SC_i = E(SM_i , k_{i,q})$). The signature message SM_i is 128 bits that is constructed by applying the MD5 algorithm on the concatenation of the DM_i, the node identification number id_i, and the $sink_q$. Finally, the data body DB_i will be formed by concatenating SC_i, DC_i and id_i as shown in Figure 7. Note that, the non-repetition is achieved by using the time stamp.

After the generation of the data body DB_i by the node n_i, the next forwarding node n_j is selected by the NNSP protocol. Next, the node n_i will construct the packet cipher PC_i which is the encrypted version of the packet message PM_i using the AES algorihm in CBC mode with k_s and IV_s (i.e. $PC_i = E(PM_i , k_s, IV_s)$, where PM_i is the concatenation of id_i, DB_i, and a new time stamp NTS_i. Finally, the node n_i sends a packet $Packet_{i,j}$ to the node n_j. The $Packet_{i,j}$ contains the source identification number id_i, the destination identification number id_j, and the packet cipher PC_i. The PC_i is generated to assure that the received message at destination node n_j is created by an authentic source node.

During the forwarding process, when an intermediate node receives a packet, it extracts the source identification *sid*, the destination identification *did*, and the packet cipher PC_{sid} and executes the algorithm shown in Figure 8.

IntermediateNodeAlgorithm (*sid, did,* PC_{sid})
1 $PM_{sid} \leftarrow D\ (PC_{sid}, k_s, IV_s)$
2 Extract id_{sid}, TS_{sid}, and DB_i from PM_{sid}
3 IF id_{sid} = *sid* & $id_{sid} \in rt_{did}$ // rt_{did} is the routing table of the node *did*
4 IF TS_{sid} matches with the time stamp of n_{sid}
// the node n_{did} should apply the following steps
5 select a node n_j form rt_{did} using the NNSP protocol
6 $TS_{did} \leftarrow$ generate time stamp matches with n_{did}
7 $PM_{did} \leftarrow DB_i \,||\, did \,||\, TS_{did}$
8 $PC_{did} = E\ (PM_{did}, k_s, IV_s)$
9 Create $Packet_{did, j}$ *did*, id_j, *and* PC_{did}
10 The node n_{did} send $Packet_{did, j}$ to the node n_j
11 ELSE
12 ignore PM_{sid}
13 END
14 ELSE
15 ignore PM_{sid}
16 END
END

Figure 8. The algorithm executed by an intermediate node.

The algorithm recovers the PM_{sid} by decrypting the PC_{sid} using the AES algorithm with k_s and IV_s in CBC mode. Then, it extracts id_{sid}, DB_i, and the time stamp TS_{sid} as indicated in Steps 1 and 2. Note that, DB_i is the original data body of the sensing node n_i . According to step 3, for the process to be continued the id_{sid} must be equal to the source identification *sid* and exists in the routing table of the current intermediate node n_{did}. Next, if the time stamp TS_{sid} matches the time stamp of the node n_{sid}, then it perform the steps 5 to 10. This ends with sending the $Packet_{did,j}$ to the next forwarding node. This algorithm is executed by all intermediate nodes until it reaches the sink node.

Once a sink node which represents the 3rd category receives a packet $packet_{sid,q}$ from an intermediate node, it extracts the source identification number *sid*, the sink identification number $sink_q$, and the packet cipher PC_{sid} and then, executes the algorithm shown in Figure 9.

SinkNodeAlgorithm (sid, $sink$, PC_{sid} **)**
1 $PM_{sid} \leftarrow D\ (PC_{sid}, k_s, IV_s)$
2 Extract id_{sid}, TS_{sid}, and DB_i from PM_{sid}
3 IF $id_{sid} = sid$ & $id_{sid} \in rt_{did}$
4 IF TS_{sid} matches with the time stamp of sid
// $sink_q$ is the identification number of the sink received the packet.
5 IF $sink = sink_q$ *and* PM_{sid} is intended to $sink_q$
6 decompose DB_i to get id_i, SC_i, and DC_i
7 $DM_i = E\ (DC_i, k_{p,i})$
8 $SM \leftarrow sink_q \| id_i \| DM_i$ *//id_i is the ID of the sensing node*
9 $SC \leftarrow$ MD5 (SM)
10 IF $SC = SC_i$ *// integrity is confirmed*
11 decompose DM_i *to get* d_i, p_s, *and* TS_i
12 IF TS_i and p_s matches those of the node n_i
13 consider data d_i
14 END
15 END
16 ELSE IF $sink = sink_p$ *and* PM_{sid} is intended to $sink_p$
// $sink_p$ is the other sink node.
17 $TS_{sinkq} \leftarrow$ generate time stamp
18 $PM_{sinkq} \leftarrow DB_i \| sink_q \| TS_{sinkp}$
19 $PC_{sinkq} = E\ (PM_{sinkq}, k_s, IV_s)$
20 *Create the* $Packet_{sinkq,sinkp}$ from $sink_q$, $sink_p$, PC_{sinkq}
21 $sink_q$ sends $Packet_{sinkq,\ sinkp}$ to $sink_p$ through the Internet
20 END
21 END
21 END
END

Figure 9. The algorithm executed by a sink node

Next, Steps 1 to 4 of the algorithm shown in Figure 9 will be executed in the same way done in Figure 8. Next, if the destination sink node is $sink_q$ and the data included in the PM_{sid} is intended for it, the $sink_q$ will first decompose DB_i to get id_i, SC_i, DC_i as described in Steps 5 and 6. Next, it recovers DM_i in step 7 and constructs its signature SC through Steps 8 to 9.

Next, the $sink_q$ will check the data integrity through comparing the constructed signature SC with the received one SC_i. If the integrity is confirmed, $sink_q$ will decompose the recovered DM_i in step 7 to get d_i, $P_{s\ ,}$ and TS_i of the original sensing node n_i. Then, if the received time stamp TS_i and packets sequence P_s match those of the node $n_{i,}$ the data d_i is considered satisfying confidentiality, integrity and refreshed data.

Otherwise, if the data is intended to the other sink node $sink_p$, In this case, the node $sink_q$ (the current sink node received the packet) will construct the packet cipher PC_{sinkq} as $PC_{sinkq} = E(PM_{sinkq}\ ,\ k_s,\ IV_s)$, where PM_{sinkq} is the message produced by concatenating the $sink_q$, DB_i, and TS_{sinkq}. Finally, the sink node $sink_p$ creates a packet $Packet_{sinkp,\ sinkq}$ and sends it to the node $sink_q$ through the internet as shown in Steps 16 to 21 of Figure 9. Thus, when the node $sink_q$ receives the packet, it will execute the same algorithm to recover the data $d_{i.}$

This is the end of the four phases of the proposed NPP protocol and the rest of this section is dedicated to demonstrate how the key k_s shared among all sensor nodes in the network can be changed throughout the network lifetime.

The main idea of changing the shared key is to generate a new shared key at regular intervals with the aid of μTESLA protocol. Our methodology starts when the sink node $sink_1$ initiates a broadcast request packet to all nodes in the network field. The packet format is in the form *"KCR.RC,"* where *KCR* refers to key change request packet and the request cipher *RC* is the encrypted version of the request message *rm* with the key k_i . The message *rm* is formed by concatenating the id_{sink1}, TS_{sink1}, and k_i, where id_{sink1} is the identification number and the time stamp of $sink_1$, respectively. The k_i used at the interval i is the i^{th} key of the key chain K generated during the initialization phase. When this packet is received by any authorized node n_j, it will store it until it receives a confirmation packet from the sink node $sink_1$ later.

After a predetermined delay time, the sink node $sink_1$ will send a key change confirmation packet in the form *"KCC.CC,"* where *KCC* refers to key change confirmation packet and the confirmation cipher *CC* is the encrypted version of the confirmation message *cm* with the current shared key k_s and the initialization vector IV_s. The message *cm* is formed by concatenating the id_{sink1}, NTS_{sink1}, and k_i, where NTS_{sink1} is a new time stamp of $sink_1$.

When a node n_j receives the confirmation packet, it extracts the key k_i from the confirmation message *cm* obtained by decrypting the confirmation cipher *CC* (i.e. $cm = D(CC, k_s, IV_s)$. Next, the node n_j checks the k_i with its commitment key stored at interval i-1 against $k = MD5\ (k_i)$. if both are equal, it means that k_i is a valid key. Next, the node n_j obtains the *rm* by decrypting the *RC* using k_i (i.e. $rm = D(RC, k_i)$. Then, the node n_j compares the k_i revealed from *rm* with the k_i revealed from *cm*. If both are equal, the node n_j starts to generate the new shared key k^*_s as $k^*_s = MD5\ (concat(k_s, k_m)$ where k_m is the master key stored for this purpose during the initialization phase. The *concat()*is a predefined concatenation function. After a predetermined delay time the node n_j considers the k^*_s as the new shared key.

IV. Security Aspects of DSSRP

This section discusses the security features of DSSRP protocol. The DSSRP protocol provides protection for WSNs against most of possible attacks on the physical and network layers. The protection against attacks such as, revealing, tampering, repeating, spoofing of data; is achieved by providing encryption, digital signature, and freshness features for the conveyed data. Also the DSSRP provides security features to guard in particular against attacks on the network layer that draw traffic by advertising high quality path to the sink node. This security feature of DSSRP is due to that in our protocol each node is permitted to receive only from and send to authentic nodes. Hereafter, we demonstrate how the DSSRP protocol can help securing the network against the aforementioned attacks.

Tampering routing information attack is defended by the DSSRP protocol when executing of the *elimination* phase. The behavior of the DSSRP protocol does not allow any node to update its routing table based on advertized information. Consequently, a malicious node cannot tamper with the routing tables of other nodes.

In selective forwarding attacks, malicious nodes drop part or all received packets so that they are not propagated any further. These types of attacks are typically most effective when the attacker is explicitly incorporated into the routing path during the data flow. The DSSRP protocol prevents a malicious node to be included on the routing path. A forwarding node randomly selects a next node towards the sink from its routing table that only includes authentic nodes.

Next, sinkhole attacks try to entice traffic from sensor nodes to the sink node. These types of attacks are hard to be defended in protocols that utilize advertized information such as Min-Hop and PEW protocols because this information is not easy to be verified. However, in DSSRP protocol, the construction of the routing path does not depend on advertized information. Consequently, an attacker employing a sinkhole

attack cannot deceive a forwarding node to send its packets to a malicious node launched this attack.

The wormhole attack uses a private and out-of-band channel between at least two malicious nodes to forward data packets from one place to another in the network through this channel. One malicious node eavesdrops data packets from its place and sends them through the established channel to the other malicious node which in turn, forwards them to a node of its neighborhood. This type of attack is detected and defended by the DSSRP protocol because in DSSRP when a node receives a data packet, it first checks whether it comes from a valid neighbor or not. If so, it will consider the packet. Otherwise, it simply drops the packet. In this case, the receiving node will drop any packet comes through the established channel.

Finally, in a spoofing attack, a malicious node impersonates a valid node. It floods its neighbors with packets that have the identity of valid IDs. In our protocol, valid data packets can only be formed by valid nodes. Thus, this attack can be launched with reply packets which are detected through the time stamp. Therefore, the receiving nodes will detect these kinds of packets and simply drop them.

V. Experimental Results

In this section, we first present the simulation environment and then demonstrate our experimental results.

A. Simulation Environment

To verify the performance of the proposed DSSRP protocol, a WSN network simulator is developed. In the simulations, the performance of the proposed protocol is compared with the PEW protocol and Min-Hop protocol (MHP) and EESRP protocol under attack-free and attack conditions.

The simulation environment is similar to that used in [4] using a random network topology. In which, 300 sensor nodes were randomly scattered in a fixed area of $50 \times 50\ m^2$, and two sink nodes are located at (0,0) and (50, 35) and both sink nodes are connected together through the internet. Fire points are chosen randomly. Sensors detecting a fire should transmit a data packet of 1024 bits to the sink node. The energy update packet size is assumed to be 64 bits for both PEW and Min-Hop protocols. Also the value of the maximum energy weight W_{max} is chosen to be 4 for the PEW protocol. In our simulation, a total of 5000 data packet were generated and fed to the four protocols, simultaneously. Each sensor node is initialized with 1000,000 units of energy (i.e., $E_{\max}$) and assumed to spend 1 energy unit for receiving and 1 energy unit for transmitting one data bit. In addition, each sensor node has a maximum transmission and detection range of 5m.

B. Simulation Results

During the course of our results, three experiments were carried out. The first experiment is performed under normal operation. The second experiment is executed under the sinkhole attack and the third experiment is operated under the spoofing or Sybil attack. The first experiment examines the performance of the four protocols under attack-free condition. The percentage of successfully received packets, the percentage of died sensors and the percentages of undetected fire points are presented for all of the four protocols in Figure 10, Figure 11 and Figure 12, respectively.

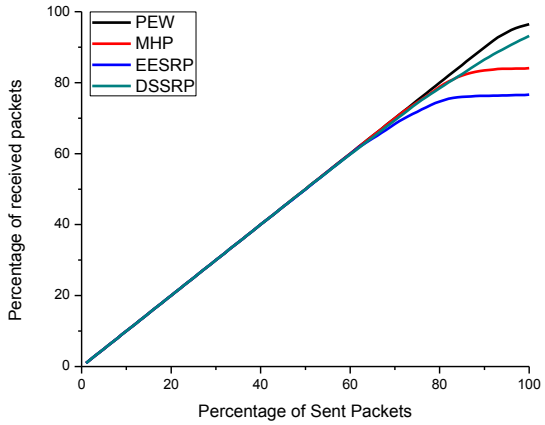


Figure 10. Percentage of received packets under attack-free condition.

The results shown in Fig. 10 indicate that, up to 70% of the sent packets, all of the four protocols have similar behavior. After the 70% of the sent packets, the EESRP protocol provides less percentage of received packets compared to other protocols due to the overhead bits that are added to each packet as a result of encryption and the addition of a hash value to provide confidentiality, integrity, authentication, and non repetition of messages. After 85% of the percentage of received packets, the MHP provides less performance compared to the PEW and DSSRP protocols. It is also worth noting that the PEW has the highest percentage of received packets compared to the other three protocols due to the global calculations of the path information. Figure 10 also indicates that the performance of the proposed DSSRP is very close to that of the PEW protocol and the difference between the percentage of successfully received packets of the proposed protocol and that of the PEW routing protocol is about 3% at its maximum. However, this sacrifice in the percentage of successfully received packets using the DSSRP is due to the security overhead.

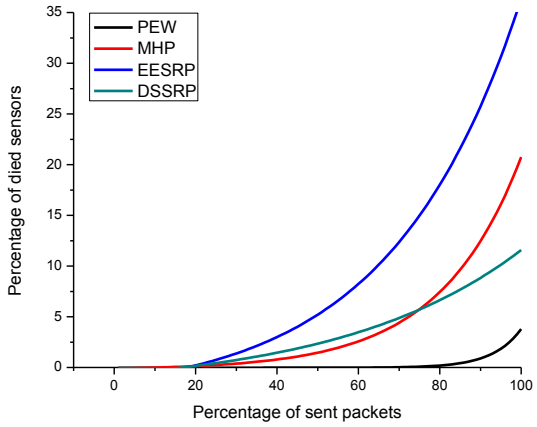


Figure 11. Percentage of died sensors under attack-free condition.

The plot in Figure 11 shows that, the PEW protocol has the lowest percentage of died sensors while EESRP protocol has the highest percentage of died sensors. On the other hand, the MHP protocol provides lower percentage of died sensors until

the percentage of received packets reaches 78% then the performance of the proposed DSSRP protocol outperforms that of the MHP. This reflects the potential of the proposed DSSRP to extend the network lifetime by providing lower percentage of died sensors especially after the percentage of sent packets reaches 78%.

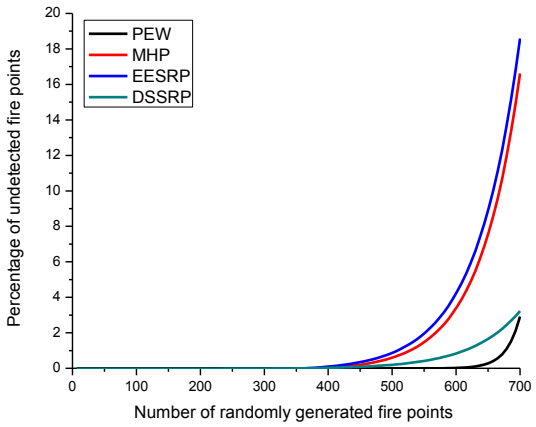


Figure 12. Percentage of undetected fire points under attack-free condition.

The simulation results shown in Figure 12 present that by randomly generating about 400 fire points all the four protocols have a similar performance. However, by linearly increasing the number of randomly generated fire points to 700 the percentage of undetected fire points by the base station using the MHP and EESRP protocols increases exponentially to reach 20%. However, with the PEW and DSSRP protocols the percentage of undetected fire points reaches about 2% at maximum.

In the second experiment, a sinkhole attack is injected into the underlying network. Similar to the first experiment, the second experiment investigates the percentage of successfully received packets versus the percentage of sent packets for each protocol as depicted in Fig. 13.

In this experiment, the sinkhole attack is implemented as follows: three unauthorized nodes were randomly added to the network of each approach. The rule of each of these fake nodes is to deceive its neighbors to forward their packets through it. For example, a fake node misleads its neighbors in the Min-Hop by updating their routing tables with the highest energy level since the protocol allows a node to update the energy levels of its neighbors. Similarly, a fake node in the PEW approach misleads its neighbors that it has the best path by updating their routing tables with the value of its PEW parameters during the update process in which each node conveys its PEW parameters to its neighbors. In these two cases, the fake nodes lured their neighbors to forward packets through them. Consequently, the fake nodes will have ability to drop these packets. On the other hand, these nodes in the proposed protocol will be detected and removed from the routing tables during the elimination phase.

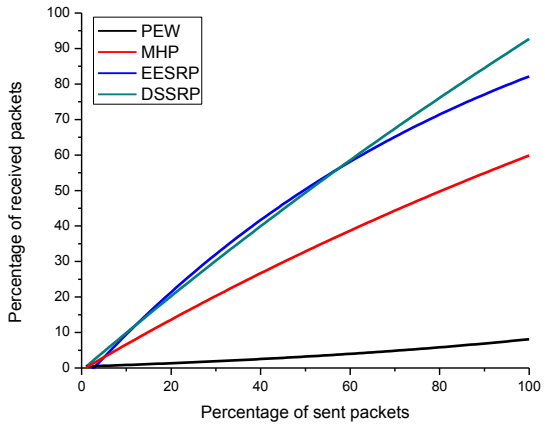


Figure 13. Percentage of received packets under sink-hole attack

The results depicted in Fig. 13 show that the performance of the EESRP and DSSRP protocols surpasses the performance of the other two approaches under the sinkhole attack. It is also clear that the performance of the EESRP and DSSRP protocols is not affected by the sinkhole attack. On the other hand, Figure 13 points out the PEW approach is the most affected protocol with the sink-hole attack and provides a very low percentage of successful packets due to the global calculations of the path information. This makes the fake path information broadcasted by the malicious nodes goes across the network and thus using these malicious nodes in forwarding most of the packets sent. Hence, most of the packets will pass through the unauthorized nodes which in turn drop them. On the other side, the performance of the Min-Hop protocol is moderate between the performance of the PEW and DSSRP protocols.

During the last experiment, a spoofing or Sybil attack is implemented to investigate its effect on the deployed network using each of the four protocols. The spoofing attack can be easily implemented by inserting one or more unauthorized nodes into the field. Actually, three unauthorized nodes are inserted; each around a corner of the three corners of the field other than the corner that contains the sink node at (0, 0). The rule of these unauthorized nodes is to continuously send malicious packets each of size 1024 bits (similar to the size of the data packet) during the network operation. The percentage of successfully received packets is presented in Fig. 14.

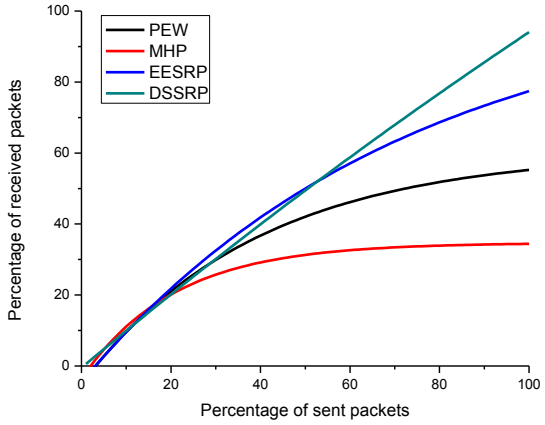
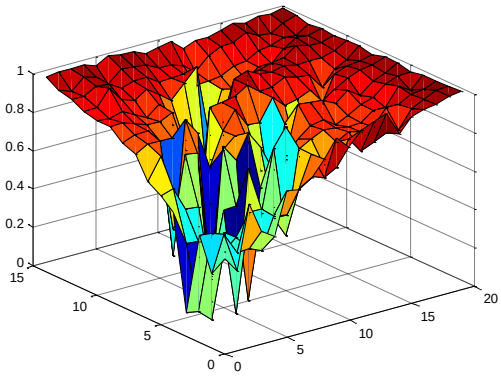


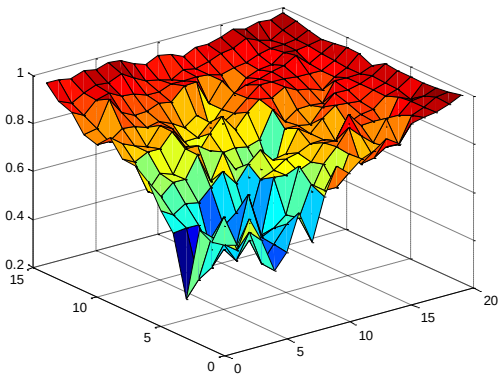
Figure 14. Percentage of received packets under spoofing attack

The results depicted in Fig. 14 clearly show that the EESRP and the proposed DSSRP protocols protect the network against the spoofing attack. On the other hand, these malicious nodes affect the Min-Hop and PEW approaches by consuming a lot of energy which shorten the network life time, especially; in a network employing the Min-Hop protocol.

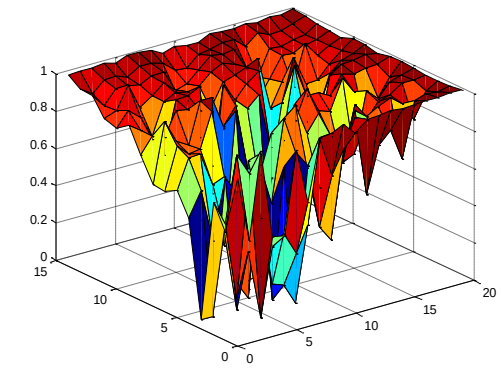
Hereafter, we investigate the energy distribution across the network for all the four protocols under attack-free, spoofing attack and sink-hole attack conditions in Figure 15, Figure 16 and Figure 17, respectively. To achieve this, the network area 50x50 m^2 is segmented into 20x15 segments (i.e., 300 segments) such that nearly each sensor node fits into a segment.



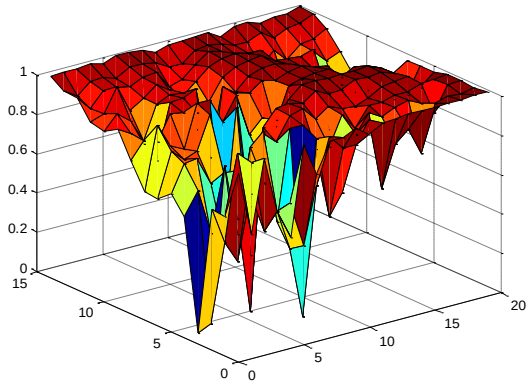
a) MHP



b) PEW

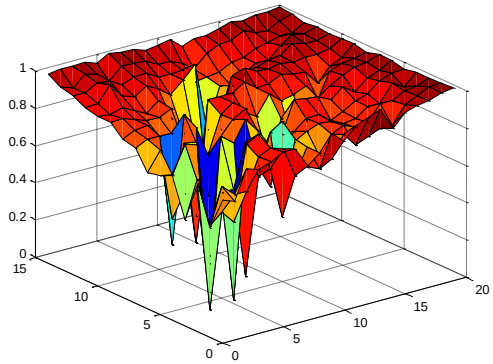


c) SRP

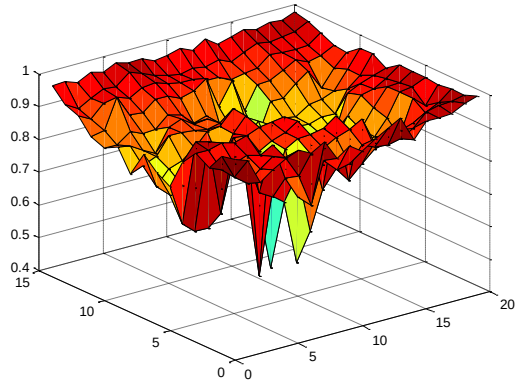


d) DSSRP

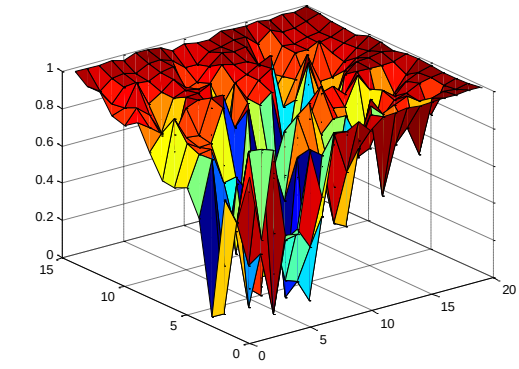
Figure 15. Normalized energy distribution under attack-free condition.



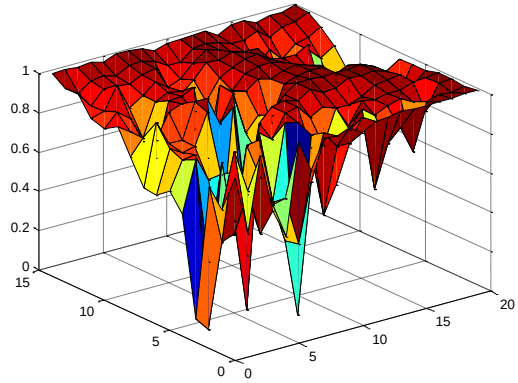
a) MHP



b) PEW



c) SRP



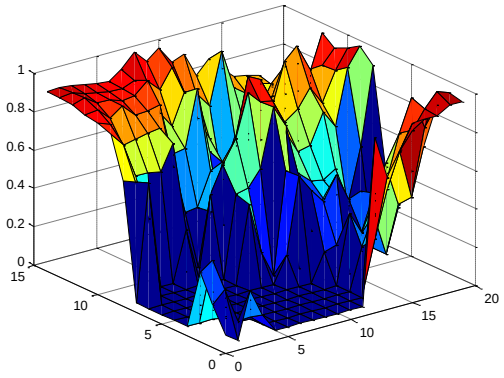
d) DSSRP

Figure 16. Normalized energy distribution under sink-hole attack.

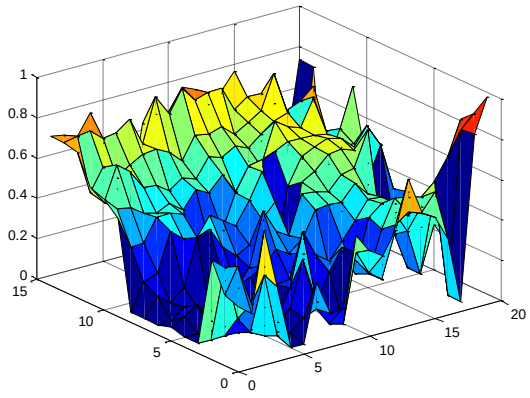
Normalized energy distribution shown in Figure 15

indicate that, under attack-free condition, the PEW protocol has the best energy distribution due to having the global energy view of the network as the main criteria in path selection. It is also clear that, the DSSRP provides a grateful energy distribution among the network nodes. However, the energy level decreases near the two sink nodes due to the frequent use of these nodes in delivering messages from all over the network to the sink nodes.

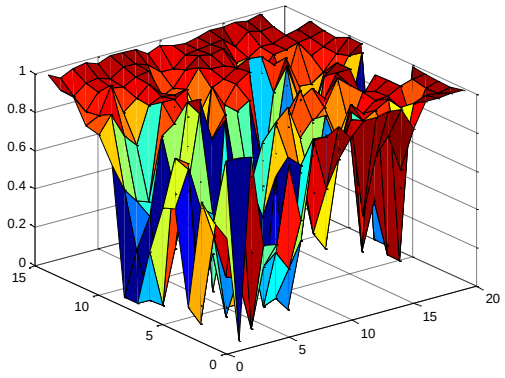
The results in Figure 16 show the normalized energy distribution under sink-hole attack. The results in Figure 16a and Figure 16b show that the normalized energy distribution along the network is higher than that of the attack-free condition. This can be interpreted as the malicious nodes have succeeded in stopping some packets from being delivered to the sink node. Hence less energy is consumed across the network. This also confirms with the result in Figure 13 where the percentage of the received packets using the PEW and MHP is relatively low indicating that great number of packets has been stopped by the malicious nodes. On the other hand, Figure 16c and Figure 16d indicate clearly that using the secure routing protocols (EESRP and DSSRP) the energy distribution is the same as that under attack-free condition (Figure 15c and Figure 15d). This shows that the secure routing protocols can successfully protect the network against the sink-hole attack.



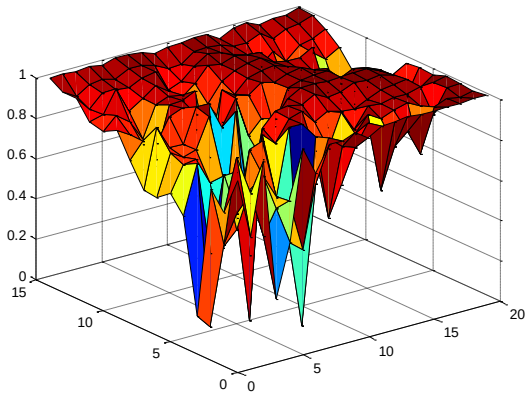
a) MHP



b) PEW



c) SRP



d) DSSRP

Figure 17. Normalized energy distribution under spoofing attack.

Figure 17 shows that, under spoofing attack, employing unsecure routing protocols (MHP and PEW) will lead to the battery depletion of several network nodes. This is very clear in Figure 17a and confirms with the results in Figure 14 where using the MHP protocol results in less than 40% of the sent packets are received by the sink node. With PEW protocol, although nodes near the sink nodes are almost depleted, the energy distribution among the network nodes is almost uniform. On the other hand, Figure 17c and Figure 17d show that, using secure routing protocols (SRP and DSSRP) protects the network against such attacks. It is also clear that, the DSSRP protocol achieves better energy distribution than EESRP.

VI. Conclusion

This paper introduces a Dual-Sink Secure Routing Protocol (DSSRP) that provides security for data packets and also prolongs the network life time by evenly distributing network load among deployed sensors. The proposed protocol enhances the EESRP protocol by increasing the number of security bits, having a variable shared key that can be changed at regular intervals, and enhancing the protocol scalability by dealing with dual sink nodes instead of a single sink node. Extensive simulation results show that, the DSSRP outperforms when compared to the PEW, Min-Hop, and EESRP protocols under various attack conditions: sinkhole attack, and spoofing or Sybil attack while exhibiting a grateful performance under attack-free condition.

Acknowledgment

This work is supported by the deanship of scientific research, Taibah University under Grant Number 715/413.

References

[1] Pandana, C.; Liu, K.J.R.; , "Robust connectivity-aware energy-efficient routing for wireless sensor networks," *Wireless Communications, IEEE Transactions on* , vol.7, no.10, pp.3904-3916, October 2008

[2] S.Chiang, C.Huang, K.Chang, "A Minimum Hop Routing Protocol for Home Security Systems Using Wireless Sensor Networks", *IEEE Transactions on Consumer Electronics*, Volume 53, No. 4, PP. 1483 – 1489, Nov. 2007.

[3] Mohamed Mostafa A. Azim, "MAP : A Balanced Energy Consumption Routing Protocol for Wireless Sensor Networks," Journal of Information Processing Systems, vol. 6, no. 3, pp. 295-306, 2010.

[4] Aly M. El-semary and Mohamed Mostafa A. Azim, Path Energy Weight: A global energy-aware routing protocol for wireless sensor networks, In the Proceedings of *IEEE* Conference on IFIP Wireless Days 2010, Oct. 2010.

[5] Pei Huang; Chen Wang; Li Xiao; , "Improving End-to-End Routing Performance of Greedy Forwarding in Sensor Networks," *Parallel and Distributed Systems, IEEE Transactions on* , vol.23, no.3, pp.556-563, March 2012

[6] Ehsan, S.; Hamdaoui, B.; , "A Survey on Energy-Efficient Routing Techniques with QoS Assurances for Wireless Multimedia Sensor Networks," *Communications Surveys & Tutorials, IEEE*, no.99, pp.1-14, 2011.

[7] Al-Karaki, J.N.; Kamal, A.E.; , "Routing techniques in wireless sensor networks: a survey," *Wireless Communications, IEEE* , vol.11, no.6, pp. 6- 28, Dec. 2004

[8] Guoxing Zhan; Weisong Shi; Deng, J.; , "Design and Implementation of TARF: A Trust-Aware Routing Framework for WSNs," *Dependable and Secure Computing, IEEE Transactions on* , vol.9, no.2, pp.184-197, March-April 2012.

[9] Wenjun Gu; Dutta, N.; Chellappan, S.; Xiaole Bai; , "Providing End-to-End Secure Communications in Wireless Sensor Networks," *Network and Service Management, IEEE Transactions on* , vol.8, no.3, pp.205-218, September 2011.

[10] A. D. Wood, L. Fang, J. A. Stankovic, and T. He, "SIGF: A family of configurable, secure routing protocols for wireless sensor networks," in *Proc. 4th ACM Workshop Security of Ad hoc Sensor Networks*, pp. 35–48, 2006.

[11] Xiangqian Chen; Makki, K.; Kang Yen; Pissinou, N.; , "Sensor network security: a survey," *Communications Surveys & Tutorials, IEEE* , vol.11, no.2, pp.52-73, Second Quarter 2009.

[12] El-Semary, A.M.; Azim, M.M.A.; "A two-tier energy-efficient secure routing protocol for Wireless Sensor Networks," *The 7th International Conference on Information Assurance and Security (IAS 2011)*, pp.331-337, Dec. 2011.

[13] Xiaobing Wu; Guihai Chen; , "Dual-Sink: Using Mobile and Static Sinks for Lifetime Improvement in Wireless Sensor Networks," *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*, vol., no., pp.1297-1302, 13-16 Aug. 2007.

[14] National Institute of Standards and Technology (NIST): Advanced Encryption Standard (AES), FIPS-197 (2001).

[15] IEEE: IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPAN), IEEE Std 802.15.4 (2003)

[16] 1ZigBee Alliance: ZigBee Specification Version 1.0 (December 2004)

[17] Ronald L. Rivest, The MD5 Message Digest Algorithm, Internet RFC 1321, April 1992.

[18] Peng Ning and Donggang Liu, BroadcastAuthentication and Key Management for Secure Sensor Networks, Handbook of sensor networks: Algorithms and Architectures, edited by Ivan Stojmenović, John Wiley and Sons, Inc., 2005.

Author Biographies

Aly M. El-Semary, He received his B.S. degree in Systems and Computer Engineering from Al- Azhar University, Cairo, Egypt in 1992, and M.S. and Ph.D. degrees in Computer Science from Tulsa University, USA in 2001 and 2004, respectively. He works for the Department of Systems and Computer Engineering, Faculty of Engineering, Al-Azhar University, where he is currently an assistant Professor. However, he is currently working as a visitor for Computer Science and Engineering College, Taibah University, Saudi Arabia. His current interests include network and computer security, sensor networks, fuzzy logic, data-mining, and neural networks.

Mohamed Mostafa A. Azim, He received the BSc degree in electrical engineering from Cairo University in 1994, the MS degree from the Technical University Eindhoven, Netherlands, in 1997, and the PhD degree in computer science from Tohoku University, Japan, in 2006. In 2008, he joined the college of computer science and engineering at Taibah University. Currently, he is the head of networks and communications systems department. He is the author of several research papers published in the most reputable journals and conferences. His current research interests are optical networks survivability, routing and security protocols for wireless sensor networks.