EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

DynTunKey: a dynamic distributed group key tunneling management protocol for heterogeneous wireless sensor networks

Ramzi Bellazreg* and Noureddine Boudriga

Abstract

Wireless sensor networks (WSNs) are being used for many applications ranging from mobile target surveillance to intelligent home networking. Due to the sensitive nature of the data transmitted by these applications, appropriate protection mechanisms are needed to prevent attackers from exploiting the weaknesses of the radio links. In this paper, we propose a novel group key management scheme called DynTunKey (dynamic tunneling and group key management protocol). This paper investigates the use of secure tunnels as a solution to improve the protection of WSNs. We propose a tunneling scheme that conforms to the security requirements of WSNs while having less computational and network overhead. We also propose a solution for a dynamic integration in the secured communication of a newly deployed sensor. A set of experiments has been conducted to assess the performance of the proposed scheme with regard to recent key management protocol and to traditional tunnels built using the IPSec protocol. We found that our protocol considerably reduces the number of transmitted messages as well as the computational load, which makes it suitable for WSNs. We tested the proposed protocol considering two models of mobility of the targets which are respectively the Random Walk model and the Gauss Markov model.

Keywords: Wireless sensor networks; Encrypted tunnels; Distributed security; Group key exchange; Threshold cryptography; Dynamic architecture; Mobility models

1 Introduction

The progress of sensing and communication technologies has motivated the proliferation of wireless sensor networks (WSNs). Tiny motes [1] connected through radio interfaces can nowadays be implemented for multiple applications including target tracking, virtual reality, and intelligent home networking. The scarcity of the computational, memory, and energy resources, as well as the native vulnerabilities of the radio transmission protocols, increases the need for security services that protect the WSN-based applications.

One of the most crucial requirements regarding the security of WSNs is authentication. This stems from the fact that radio links are, by nature, open and vulnerable to various identity spoofing attacks. Moreover, the confidentiality of the gathered data is often an important concern.

*Correspondence: ramzi_bellazreg@yahoo.fr
Communication Networks and Security Research Laboratory, Higher School of Communication, University of Carthage, Carthage, Tunisia

Secure tunnels have been widely used in traditional wired and wireless networks to guarantee confidentiality and authentication. Unfortunately, the use of traditional tunneling protocols, mainly IPSec and SSL, is not suitable with the specific features of WSNs. Most of the research published in the literature has focused on the development of light cryptographic algorithms that comply with the sensor node capabilities. However, tunnel management protocols and the underlying key handling schemes have not been addressed. The objective is to present a solution that permits the following:

- The authentication of the sensing nodes
- The formation of *ad hoc* secure channels for every cluster
- The security of the exchanges between the nodes

To ensure those objectives, we propose in this paper a distributed and dynamic tunnel and group key management protocol for WSNs called DynTunKey. We introduce a tunneling approach that takes into account the characteristics of the cryptographic algorithms that are typically used for WSNs. The most important contributions of the solution presented are listed in the following:

- The proposed approach adapts to heterogeneous WSNs. In other terms, instead of using 1-to-1 tunnels, we rather build many-to-many tunnels. We will have a unique tunnel for many sensors communicating together. That is advantageous because we will ensure several communications between multiple nodes through the same security tunnel.
- In our work, we introduced a new concept which is the cluster security association (CSA). The CSA is an abstraction of the established many-to-many tunnels and represents shared security attributes between many sensor nodes.
- In the proposed approach, the tunnel key used for the encryption of the communicated data will not have an infinite validity time and will be changed periodically. This characteristic is advantageous and enhances the robustness of the security of the protocol and protects against node compromise.
- In the proposed approach, the key is not predeployed at the sensors but is dynamically generated. The communicating sensors contribute in a secure manner in this generation. Then, we avoid a full centralized management scheme
- The proposed protocol provides for a dynamic integration of new sensor at any time to the global architecture without compromising the security needed. A new appearing sensor is automatically inserted into the communicating nodes without the need of updates in the other nodes.

Then, the proposed protocol permits all the sensors to exchange directly secured data. This is useful in many applications that necessitate the exchange of data between all the sensors. In particular:

- Military target tracking application needs such communication. When using a group key, each sensor can report its gathered data using the group key to the other sensors. This is advantageous and permits collaboration between the sensors without the need of central node.
- Firefighting applications. In this case, the sensors are deployed with the firefighter. If all the sensors have the same shared key, any one of the sensors can directly send data to other sensors. Then, the collaboration between the members of the team will be easier by having an efficient dialog between the sensors that represents the firefighter.

The rest of the paper is organized as follows: In Section 2, we will present some of the most important security solutions and key distribution schemes used for wireless sensor networks. Section 3 describes the global security architecture that will be used to protect heterogeneous WSNs. The tunnel initialization phase is addressed in Section 4. A protocol for distributed negotiation and management of the cluster security associations required to establish the many-to-many secure tunnels is introduced in Section 5. Then, we present a method for dynamic integration of new sensors in Section 6. In Section 7, we will analyze the robustness of the proposed tunneling protocol. Section 8 presents the simulation model including the deployment of the sensors and the mobility models for the targets. In the same section, we present the results of some simulations to compare our protocol to recent key management solutions for wireless sensor networks. The same section assesses the efficiency of the proposed protocol and compares it to the classical tunneling IPSec approach in terms of communication and processing overhead. Finally, Section 9 concludes the paper.

2 Related works

In this section, we will present the most cited security solutions used for the WSN. We especially focus on the key management side of each protocol. For each solution, we will present the advantages and contributions of the proposed protocol when compared to this solution.
The first proposed solutions propose a straight pairwise private key sharing scheme between every pair of nodes [2-4]. In a deterministic manner, every node shares a key with every other sensor, and the communications are done over 1-to-1 secured tunnels. When compared to the needed security requirements in the architecture considered, those solutions present the following lacks:

- The first problem with that solution is that $n - 1$ keys have to be stored at each sensor node, where n is the number of the nodes in the network. Thus, a large memory space is used to store all the keys. For our solution, only one key is used between a set of sensors that belong to the same group.
- A compromise of a node will compromise its communication for all the network, because this sensor node stores in its memory all the network keys. This is not the case in our solution because the key is always renewed.
- The other major lack of this solution is that it does not permit a simple group communication. In fact, if a sensor needs to send data to many sensors, it has to do this in many messages. Many copies of the message have to be sent directly to the sensors using the pairwise shared key with each one of the sensors.

This is impractical when the number of the sensors belonging to the group becomes larger. For our solution, we have only one key shared with all the sensors. According to this, a node sends the data only one time using the shared group key.
- Another problem with this solution is the addition of a new sensor to the list of the communicating node. In fact, in those solutions, all the keys are predeployed in the sensors. Then, to add a new sensor, we have to update the keys stored in all the sensors to add the key that will be used with the new sensor. For our solution, we will show that a new sensor is dynamically integrated in a secured communicating group.

Other probabilistic solutions were presented and used for WSNs. Those solutions are qualified as random key distribution solutions. In the basic random key scheme, Eschenauer and Gligor [5] introduced a probabilistic key predistribution scheme for sensor networks. This solution is based on three steps which are respectively the key predistribution, the determination of the shared key phase, and the path key establishment. In the key predistribution phase, initially a large pool of P keys are chosen, and each sensor will be equipped with a key ring stored in its memory. The key ring consisted of randomly chosen keys from the set P. Then, the neighboring sensors will find which is the common key in their rings. This key will be used to secure the data sent between the two sensors. This is done in the shared key phase. The last phase is the path key establishment phase. In fact, this solution is probabilistic because it is not guaranteed that all the combination of the pair nodes shares a common key in their randomly chosen rings. Then, a path key has to be assigned for those sensor nodes through two or more links established at the end of the second phase. This solution when compared with the previous solutions necessitates less amount of memory in the sensors because each sensor does not store a key with all the other sensors.

Inspired by this work, additional random key predistribution schemes have been proposed in [6-11]. The main addition of those works is to increase the resilience of the network against node capture and ensure a smaller need for communication intermediate paths. Those solutions also optimize the required operation time and the number of the stored keys. However, despite all the added techniques, it is always a probabilistic solution. The major lacks of those solutions for our context are presented in the following:

- The previous kind of solutions, i.e., those that store a lot amount of keys in the memory of each sensor
- This solution does not permit direct group communication between sensors, because the links established are 1-to-1 links. Also, those links are not directly established because its impossible to find a shared key between all the pairs of nodes. Then, if a sensor needs to send data to a group of nodes, it has to do it in a separate manner for each sensor.

Another category of the key generation solutions is the centralized key management schemes. In those schemes, a central node called the key distribution center (KDC) controls and generates the keys used by the sensors. One of the protocol functioning in this manner is the LKHW protocol proposed in [12]. In this scheme, the core node is treated as a KDC, and all keys are logically distributed in a tree rooted at the base station:

- In this solution, the sensors does not contribute in the elaboration of the keys. Then, a compromise of the central node compromises all the network chain.
- Another lack in this solution is that the keys are distributed in a tree manner. Then, to have a communication between a set of nodes, we do not certainly have a direct secure link between them. Then, a group communication is difficult to be proceeded in this schemes because it will be done as many separate secured connections in a tree communication manner.
- Having keys distributed in a tree manner does not facilitate the regeneration of the keys and the integration of a new sensor node in the communicating trusted group of sensors.

In PIKE [13], Chan and Perrig propose a solution that is not a fully centralized solution. The basic idea in PIKE is to use sensor nodes as trusted intermediaries to establish shared keys between nodes. In this solution, they proposed that the key will be established between two sensors through a common trusted third node somewhere within the sensor network. For this solution, initial keys are distributed such as for any two nodes A and B, there is a node C that shares a key with both A and B. Therefore, the key establishment protocol between A and B can be securely routed through C. In this solution, the establishment of the key is secured, and the number of initially deployed keys at the sensor is less than the previous solutions. However, it is not suitable for group communication, because it is least probable that all the nodes of the same group have a common trusted node. The same lacks of the previous categories of solutions are present with this kind of solution.

The LEAP protocol described by Zhu et al. [14] takes an approach that utilizes multiple keying mechanisms. In this scheme, four kinds of messages are established between the different types of sensors:

- An individual key shared with the core node (predistributed)
- A group key that is shared by all the nodes in the network (predistributed)
- Pairwise keys shared with immediate neighboring nodes
- A cluster key shared with multiple neighboring nodes

LEAP protocol permits several kinds of communications depending on the needed communicating nodes. This solution provides a many-to-many tunnel protocol like our proposed solution. However, when compared with our solution, it has some lacks:

- The number of the deployed keys at the sensors is large since every pair of sensor nodes needs a key.
- The keys used for several kinds of communication are predeployed into the sensors. This solution uses a static key and does not propose a dynamic generation of the key. For the solution we proposed, the key is renewed after a validity interval.
- In this solution, the keys used for cluster communication are predefined. Then, this solution does not permit a dynamic belonging to the groups. When a sensor needs to change from a group, the key stored in its memory have to be updated and replaced by the cluster key of the new group. That solution is not practical because it needs direct static intervention with each group change. For the proposed solution, the group key is regenerated automatically at periodic times. Then, if a sensor changed from a group, it is automatically integrated in the secured new group when it contributes to the elaboration of the group key.

The protocol NSKM presented in [15] is a protocol that manages different kinds of keys such as the LEAP protocol [14]. The difference is that the cluster keys are calculated by every node within a particular cluster. Despite this change, the main lacks of the LEAP protocol are the lacks of this protocol, in particular the absence of a rekeying solution.

In [16], the authors propose a solution called real-time dynamic key management (RDKM). The main feature of this solution is that it establishes a real-time rekey mechanism based on the search-triggered splay tree architecture. It designs and realizes the rekey mechanism based on the splay tree, which can provide random function to generate new keys and make the dynamic key management feasible. In this solution, the cluster heads organize the keys of their members into a splay tree architecture key pool. The cluster head shares with each one of the member nodes belonging to its cluster a pairwise key. Those keys are established through messages shared between the sensors and the corresponding cluster heads. This solution presents an efficient storage and rekeying solution, but it does not ensure direct group communication between several sensors because the sensors do not share a unique cluster key.

In our recent work [17], we presented a tunneling protocol adapted for wireless sensor networks. In this work, we analyze more security cases such as dynamic integration of newly appeared sensors. We also considered more simulation cases and scenarios such as the deployment and movement of the monitored events. Table 1 illustrates the comparison between the proposed solution and the previously cited solutions.

3 Architectural issues

In this section, we discuss the architectural aspects related to the implementation of encrypted tunnels on a WSN infrastructure. We first emphasize the need for using protected tunnels in the particular context of heterogeneous WSNs. Then, we provide a global overview on the proposed distributed security approach. Finally, we present the communication exchanges.

3.1 Need for encrypted tunnels in WSNs

Sensor networks can be classified into two categories: simple (or flat) sensor networks and heterogeneous sensor networks. In a flat WSN, all the sensor nodes have the same sensing, communication, and processing characteristics. A heterogeneous WSN integrates various sensor types with different capabilities. The presence of heterogeneous nodes (i.e., nodes with an enhanced energy capacity or communication capability) in a sensor network has the advantage of increasing network reliability and lifetime. Typically, a large number of inexpensive nodes perform simple sensing tasks, while a few expensive nodes (that may be embedded on mobile platforms) provide data filtering, fusion, and transport. This segregation of roles promotes a cost-effective design of the network as well as a more efficient implementation of the overall sensing application.

In this paper, we consider the particular case of a heterogenous WSN represented in Figure 1. The network is composed of two layers, the core layer and the sensing layer:

- The core layer includes nodes which are equipped with powerful sensing and transmission capabilities. Hereinafter, these nodes will be referred to as core nodes. They are able to acquire and exchange voluminous high-resolution data related to the events detected by the low-level sensors. Moreover, this layer constitutes a communication backbone allowing to spread data collected by elementary sensors on a wide area.

Table 1 Comparison of key management solutions

Protocol	Hierarchy keys per sensor	Distribution mechanism	Number of keys	Group communication	Regeneration of the keys
Probabilistic solutions [2-4]	Pairwise keys	Static predeployed	One key with every sensor	Many tunnels per group	No
Random key predistribution schemes [5-11]	Secured paths through trusted sensors	Static predeployed	One ring pool of keys (selected randomly)	Many tunnels per group	No
LKHW [12]	Tree-based key management	Static predeployed	One key with the superior node in the tree	Many tunnels per group	No
PIKE [13]	Secured paths through trusted sensors	Static predeployed	One ring pool of keys (equal to the number of needed trusted nodes)	Many tunnels per group	No
RDKM [16]	Tree-based key management	Computed through exchanges	One key for initial exchanges, pairwise keys, and one cluster key in each sensor	Many tunnels per group	On demand (one sensor departure or attack on a secure channel)
LEAP [14] and NSKM [15]	Many kinds of keys (pairwise and group)	Static or computed on predeployed	One key for group communication + one key with every sensor	One tunnel per group	No
The proposed protocol DynTunKey	One group key	Computed on exchanges	One key for initial exchanges and one group key in each sensor	One tunnel per group	Periodic

- The sensing layer consists of miniature devices, also referred to as elementary sensors, whose role is limited to the following: (a) collecting information about presumably malicious objects, (b) generating real-time events related to the detected targets and transmitting the events towards the closest core sensor, and (c) relaying the events generated by other sensors to the core sensors.

In hostile scenarios, relaying of critical data must be secure. Since data would be relayed through many nodes, care must be taken to ensure that the data aggregated at intermediate node is not corrupted. When receiving an alert message, the core node should also accurately verify the identity of the originating sensor node. In fact, the adversary can deploy sensor nodes that can deliver false information to the analysis center. Moreover, the legitimate sensor nodes are prone to be corrupted, because of weak physical protection, so as to be under the control of the enemy. Therefore, it appears that authentication and confidentiality are among the most crucial security properties that should be fulfilled when implementing a

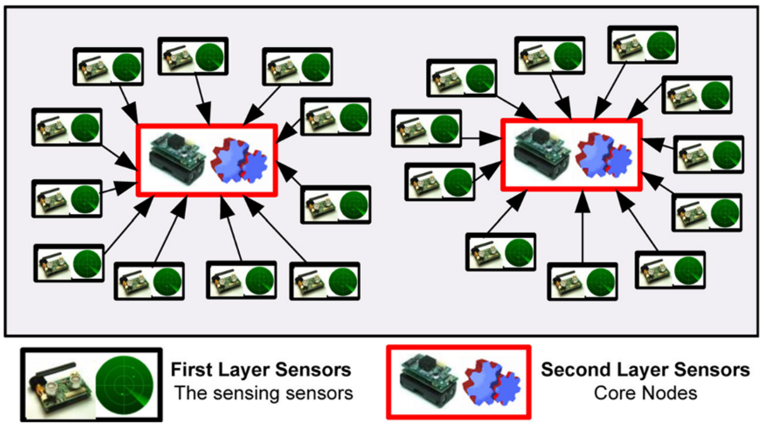

Figure 1 The network architecture.

hierarchical infrastructure. Encrypted tunnels constitute a promising alternative to address these needs since they have been widely used in many contexts in traditional networks.

3.2 Proposed security architecture

At the foundation of our approach is the idea of assembling the verification operations of the alert messages originating from multiple sensor nodes to a unique verification step. Based on this reasoning, we define a new requirement called k-security [18]. A WSN is called k-secure if, and only if, the following properties hold:

1. Every sensor node s_i possesses a private key denoted by κ_i.
2. A unique public key π and a subsequent algorithm can be used to verify whether k signatures of the same message generated by distinct sensors are valid or not.
3. An event detected by the sensing layer is considered valid at the core layer if, and only if, k corresponding alert messages are received and successfully verified (as defined in item 2) by the core layer.

The key characteristics of our work are listed in the following:

1. The proposed protocol share a unique group key between a set of sensors that belongs to the same geographic zone. Then, all the sensors can exchange data between them in a secure and simple manner. Every sensor can send secured data either to the core node or to the other sensors because the group key is known by all the kinds of sensors.
2. The proposed key generation method includes multiple phases that can be organized according to the context in which the WSN is implemented. To implement this functionality, we consider the group Diffie-Hellman key exchange protocol introduced in [19].
3. We will no longer use asymmetric encryption because the exchanged tunnel group key is a symmetric key. This is advantageous because the use of the symmetric encryption needs less resources than asymmetric encryption. It is then well adapted to the context of the wireless sensor networks.

The basic steps of the proposed security scheme are given below:

1. The core node periodically sends messages to the sensor nodes asking for new information. The sensors that have gathered new information will send reply messages.
2. The core node builds sensor clusters based on the location of the detected events, in the sense that a cluster will include the sensor nodes that have detected the same event.
3. The CSA is set up for every cluster. The tunnel establishment process is authenticated using threshold public key cryptography. The nodes of the cluster share the same symmetric group key using group Diffie-Hellman key exchange.

3.3 Communication exchanges

For our work, many messages will be exchanged between the core node and the sensors. We have two major kinds of messages. The first ones are the messages required to establish the tunnels. The second kind of messages is used to report the events detected by the sensors. The two kinds of messages are sent periodically. We divided the time into slots. At the end of each slot, the sensors send the gathered data to the core node. This reported information is encrypted using the symmetric key shared and established between the core node and all the sensors. The group symmetric key used has a validity interval which is equal to N time slots. In this manner, we will not generate a group key in each reporting slot, but we will generate a key used in many time slots. This aims to decrease the time required for the establishment of the keys. This is illustrated in Figure 2.

In the following sections, we describe the protocols that we have designed to support the implementation of the aforementioned security process.

4 Initial and authentication exchanges

The communication begins with initial exchanges. This part is composed of five steps and three types of exchanged messages. At the end of those steps, a symmetric key for the group is calculated. For each tunnel, all the nodes contribute to the calculation of this group key. This key is denoted group key Diffie-Hellman (GKDH). When all those messages have been exchanged, the sensors and the core node are authenticated using their private keys. The method used to establish this key is derived from the Diffie-Hellman key agreement protocol [20] and the work of Augot et al. [19] which presents a method to generate a key for a group.

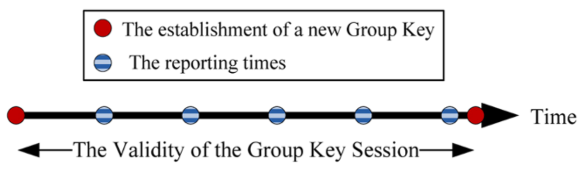
The establishment of a new Group Key
The reporting times
Time
The Validity of the Group Key Session

Figure 2 The communication exchanges.

Initially, all the nodes agree on a cyclic group G and on two numbers p and g, where p is a prime number and g is a primitive root modulo p. The nodes have also a pre-deployed value S which is secret and known only by the trusted sensors.

- *Step 1.* At this step, the first message INIT is sent by the core node to all the nodes in its coverage area to announce the beginning of an initial exchange phase. This message is sent periodically. By the nature of the wireless sensor network, the message is broadcasted to all the sensor nodes.
  {Type_of_message, INIT}
  Moreover, the core node picks a random natural number Rc. This number is the contribution of the core node in the group generated key.
- *Step 2.* The nodes that have detected some events in the last period will participate to the construction of the CSA. When receiving the INIT message, every sensor node Si picks a random integer Ri. Then, the sensor node calculates $g^{S*\mathrm{Ri}}$, joins its identifier to this value, and sends the whole message to the core node. In this message, the sensor also sends its digital signature. The field AUTH is a digest of the message and is signed by the sensor using its private key KSi. This ensures the integrity and the authentication of the sensor node.
  {Type_*of*_message, [Identifier,g^{s*Ri}]}, $\{\mathrm{AUTH}\}_{KSi}$
- *Step 3.* At this step, the core node has received the contribution of all the sensors. For each message, it verifies the integrity and the authenticity of the originating sensor node using the common public key π. When verifying the identity of the sensor nodes, only the trusted sensors will contribute to the elaboration of the group key. If an intruder tries to send a contribution, it will not be authenticated and then its contribution will not be considered and it will be rejected from the group.
  Based on the identifiers of the sensors and relatively to their deployment positions, the core node will organize the sensors into groups. Each group is the set of the sensors that detected the same event that occurs in the same zone. For each group, a common tunnel and a group key will be calculated.
- *Step 4.* At this step, the core node has received the contribution of all the sensors and classified them into groups. If the core node classified the sensors into N groups, then the core node will perform the following tasks for each group in a separated manner. For each value received $g^{S*\mathrm{Ri}}$, the core node computes the resulting value $(g^{S*\mathrm{Ri}})^{\mathrm{Rc}} = g^{S*\mathrm{Ri}*\mathrm{Rc}}$. Then, the core node sends to all the sensors belonging to the same group those values with the identifiers of the sensors. The message sent is represented below:
  {Type_of_message, [Identifier of S1, $g^{S*R1*Rc}$],..., [Identifier of Si, $g^{S*Ri*Rc}$],..., [Identifier of Sn, $g^{S*Rn*Rc}$]} , $\{\mathrm{AUTH}\}_{KCN}$
  The payload AUTH is the digest of the message. This digest is encrypted using the private key of the core node KCN. The payload AUTH and the signature ensure both integrity of the message and authentication of the core node.
- *Step 5.* At this step of the exchanges, each sensor can calculate the group key for the cluster to which it belongs. When receiving the previous presented message, each sensor verifies the authentication to check if the message is sent by the core node or by an intruder; this is done using the public key of the core node. The sensor checks also the integrity of the message to be sure that it has not been modified. After those checks, every sensor node looks for its identifier and takes the calculated value corresponding to it. For example, the sensor Si will consider the value $g^{S*\mathrm{Ri}*\mathrm{Rc}}$ and removes its secret value Ri from it to obtain $g^{S*\mathrm{Rc}}$. Then, it calculates the group key which is done by the following equation:
  $\mathrm{GKDH} = g^{S*\mathrm{Rc}} * \left(\prod_{i\in M\neq(\mathrm{CN})} g^{S*\mathrm{Ri}*\mathrm{Rc}}\right) = g^{S*\mathrm{Rc}(1+R1+...+\mathrm{Ri}+...+Rn)}$.
  This operation is performed by every sensor and then all the sensors have the same shared key. The core node also calculates the group key GKDH which will be the session key for that group of the sensors.
  Now, the initial exchange has finished. At the end of those exchanges, we have two major results. First, those messages authenticate the core node and the sensors and then only the trust nodes will participate in the establishment of the secure channels. Secondly, after those exchanges, the nodes have shared a secret group key that will be used in further communications and sessions. Those initial exchanges are illustrated in Figure 3. In this figure, the sensors S1, S2, and S3 have detected the same events, but the sensor S4 does not detect any event. When receiving the INIT message, the sensor S4 does not send a response, but the other sensors exchange all the messages and complete all the steps from step 1 to step 5 to establish the group key.

5 Cluster SA negotiation exchanges

5.1 Sensor clustering

Now that all the sensors have established the group key, a second phase is performed by the core node to identify the sensors that will participate to the CSA and their cryptographic preferences. This part is decomposed into two messages exchanged between the core node and the sensors. Those steps are performed for each group of sensors.

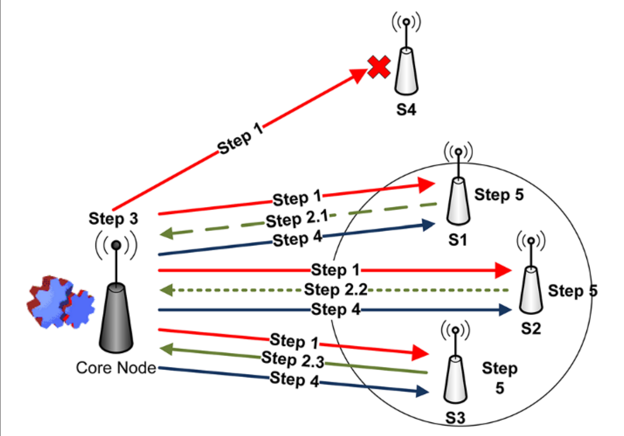

Figure 3 The group key establishment.

- *Step 1.* At the first step, the core node sends a message to all the sensors. In this message, the core node demands to the sensors to choose their preferences for the CSA. It sends the cryptographic suites supported by it. The message sent is in the following format:
  {Type_of_message, $\{\text{Mid, SAi}\}_{GKDH}$}, $\{\text{digest}\}_{GKDH}$
  Type_of_message indicates the type of the sent message. For this case, the node who receives the messages detects that it is a request to choose a cryptographic suite. Mid is the message identifier. SAi states the cryptographic algorithms supported by the core node for encryption and signature. The previous two payloads are encrypted using the previously negotiated key GKDH. This ensures the confidentiality of the transmitted data. digest is a digest of the global message and is encrypted using the negotiated key GKDH. This payload ensures the integrity and authentication of the sent data.
- *Step 2.* At this time, all the sensors have received the supported algorithms sent by the core node. Each sensor decrypts the received message using the symmetric key GKDH and then verifies the integrity of the message using the same key. Every sensor chooses its preferences of cryptographic suites and responds by sending a message in this format:
  {Type_of_message, $\{\text{Mid,ID\_Sensor,SAr}\}_{GKDH}$}, $\{\text{digest}\}_{GKDH}$
  Mid is the message identifier sent by the core node in the previous message. ID_sensor is the identifier of the sensor which sends the response. SAr states the cryptographic suite chosen from the offered choices sent in the payload SAi. These three payloads are encrypted using the group key GKDH. digest is a digest calculated and encrypted using the key GKDH.
- *Step 3.* At this step, the core node has received all the responses from the sensors and performs some tasks:
  – It decrypts the message using the symmetric group key GKDH.
  – It calculates the digest of the sent payload.
  – It decrypts the payload $\{\text{digest}\}_{\text{GKDH}}$ using the key GKDH.
  – It compares the calculated digest and the received one to verify the integrity of the message.

  If those tests are positive, the sensor is then authenticated and can participate to the CSA.

5.2 Establishment of the CSA

In the previous steps, the core node has received the identifiers of all the sensors and the selected cryptographic suites. The core node can establish the CSA corresponding to the responses. The CSA of each group of sensors contains the following information:

- The list of the sensors
- The key of the session GKDH already established in the initial exchanges
- The cryptographic suite (for signature and encryption)

The characteristics of the CSA (despite the key) are then sent to all the sensors and are encrypted using the previous group key GKDH. The payload $\{\text{digest}\}_{\text{GKDH}}$ ensures the integrity and authenticity of the core node. The message sent is represented below:

{Type_of_message, $\{\text{list of sensors, cryptographic suite}\}_{GKDH}$}, $\{\text{digest}\}_{GKDH}$

At this final step of exchanges, each sensor node has the cryptographic suite, the list of trusted sensors, and the session key. An illustration of the steps of the CSA establishment protocol is represented in Figure 4.

6 Dynamic integration of the sensors

In this section, we will present a solution to integrate a new appearance of a sensor in the list of the communicating sensors. We mean by the appearance of a sensor that it appears in an area after the beginning of the group key generation. We have two cases of appearance:

- The first one is that the sensor is already deployed in another area and has moved from a group area to another one.
- The second case is when the sensor is a newly deployed sensor and was not in another area. For the last case, we will present two strategies of integration. The first one does not take into account the time of the new sensor deployment. The second strategy of integration depends on the time of the sensor deployment.

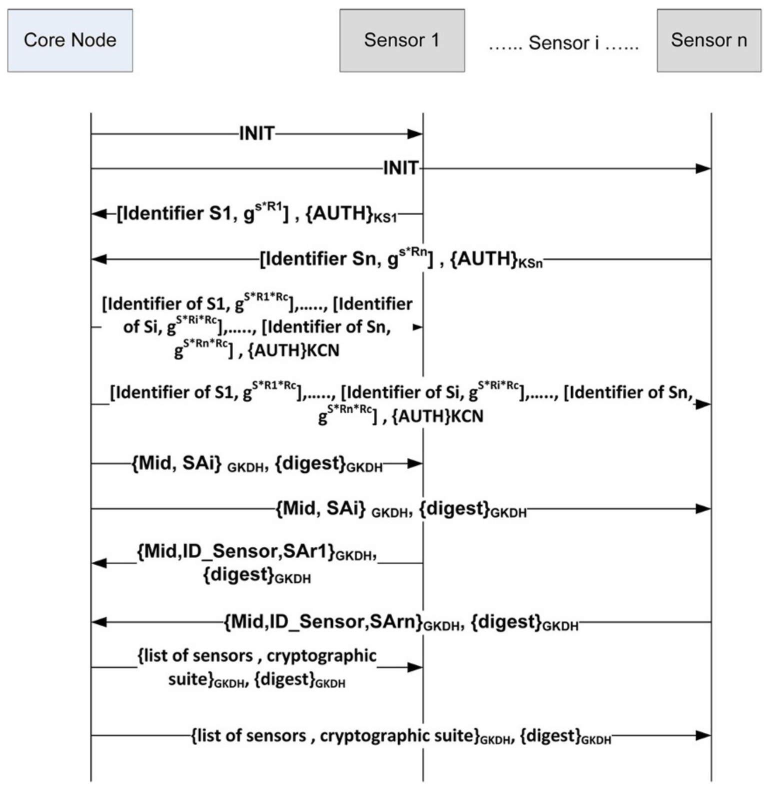
Core Node
Sensor 1
...... Sensor i ......
Sensor n
INIT
INIT
[Identifier S1, g^s*R1] , {AUTH}KS1
[Identifier Sn, g^s*Rn] , {AUTH}KSn
[Identifier of S1, g^S*R1*Rc],....., [Identifier of Si, g^S*Ri*Rc],....., [Identifier of Sn, g^S*Rn*Rc] , {AUTH}KCN
[Identifier of S1, g^S*R1*Rc],....., [Identifier of Si, g^S*Ri*Rc],....., [Identifier of Sn, g^S*Rn*Rc] , {AUTH}KCN
{Mid, SAi} GKDH, {digest}GKDH
{Mid, SAi} GKDH, {digest}GKDH
{Mid,ID_Sensor,SAr1}GKDH, {digest}GKDH
{Mid,ID_Sensor,SArn}GKDH, {digest}GKDH
{list of sensors , cryptographic suite}GKDH, {digest}GKDH
{list of sensors , cryptographic suite}GKDH, {digest}GKDH

Figure 4 **The exchanged messages.**

6.1 Integration of a previously deployed sensor

In this case, the sensor is already deployed and belongs to an initial area. The sensor has also contributed in the elaboration of a group key shared with other sensors and with the core node. In this case, due to the mobility of the sensors, this sensor has moved to another area. This sensor can continue sending its data using the shared group key relative to its initial group. At the next group key generation point, this sensor will perform the exchanges with the core node. That one will automatically assign the sensor node to its current group, and the key will be shared with the sensors that are in its area.

6.2 Integration of a newly deployed sensor

As we mentioned previously, the core node and the sensors will share a symmetric key that will be used for the encryption of the reported events. However, this key has a validity of N slots and is established only in the beginning of this interval. Thus, when a new sensor is deployed in the middle of this key's validity interval, it has not contributed in the establishment of the shared key and then cannot communicate with the core node. In this section, we will present a solution to integrate this sensor node in the communicating sensors to have a dynamic architecture.

When the sensor node Sn is newly deployed, it detects the events that occur in its coverage area. At the reporting time, it cannot send the data to the core node because the session key is unknown for him. The node Sn sends the data gathered to the neighboring node. This data is sent encrypted with the private key of the sensor node. The node Sn joins its devise identifier to the event information. This permits the identification of the sender and then its public correspondent key used for decryption. The Sn's neighbor sensor does not decrypt the data but only relays it to the core node. Thus, the intermediate sensor node sends its gathered data and the data relayed from its neighbors.

When receiving a message from a sensor, the core node will decrypt the data sent by the sensor using the group key used for the session. For the data relayed from the other sensors which are not in the group, the core node determines the identifier of the sender and decrypts the data using the correspondent public key.

Functioning in this manner, a newly deployed sensor will be integrated in the global architecture through its neighbors. When a new cycle of group key will began, the new sensor Sn will participate in the elaboration of the group key and will send directly its data to the core node encrypted with the group key. This is illustrated in Figure 5.

Integrating in this manner the new sensor, the data detected by the sensor Sn and reported by a sensor Si at the slot number n is really detected at the slot $n-1$. In fact, only at the beginning of each slot time there is a report of events by the sensor to the core node. Then, the new sensor Sn will send its data to the neighbor Si at the slot $n-1$. However, the sensor Si has already sent at this time its data to the core node. The received data from the neighbors is temporarily stored in the buffer of the sensor, and in the next reporting time n, it will join this data to its reporting information and sends the whole message to the core node. The core node must synchronize well the information. For the data detected by the sender, it is reported in its time, but for the relayed data, the core node has to consider that those events are detected in the previous slot time but are reported later cause of the time required for transfer through the intermediate node.

6.3 Time-dependent integration of a newly deployed sensor

As presented previously, a node that has been deployed after the beginning of the group key generation will be integrated by giving it the possibility of reporting its data to the core node through a neighboring sensor. The data relayed by the neighbor is always shifted by at least one slot time. Thus, if a sensor node is newly deployed at the end of the group key interval, we propose that it does not relay its data to its neighbor. In fact, reporting the data necessitates additional treatments such as neighbor discovery, relay of data to the intermediate sensors, and complex treatments at the core node. Thus, if a group key exchange interval is close, we can avoid complex treatments, and the sensor node collects and stores its data without sending it. When it contributes to the new group key generation, it will send all the data directly to the core node through the new tunnel at the first reporting slot time.

Based on those facts, we propose that if a sensor node is newly deployed in the first half of the 'group key validity interval' it reports its data to a neighbor and we avoid a big lateness of reporting events. However, if it is deployed in the second half, it can wait the elaboration of the new key to report its data. In that last situation, the data will not be reported very late, and we avoid the complex previously cited treatments. This mechanism is illustrated in Figure 6. In this figure, we have two newly deployed sensors Sn and Sn2. The sensor Sn is deployed in the first half of the interval and then it sends the detected events to the neighbor node. However, for the sensor Sn2, it is deployed in the second half of the interval and then it stores its data and sends it in the next group key interval.

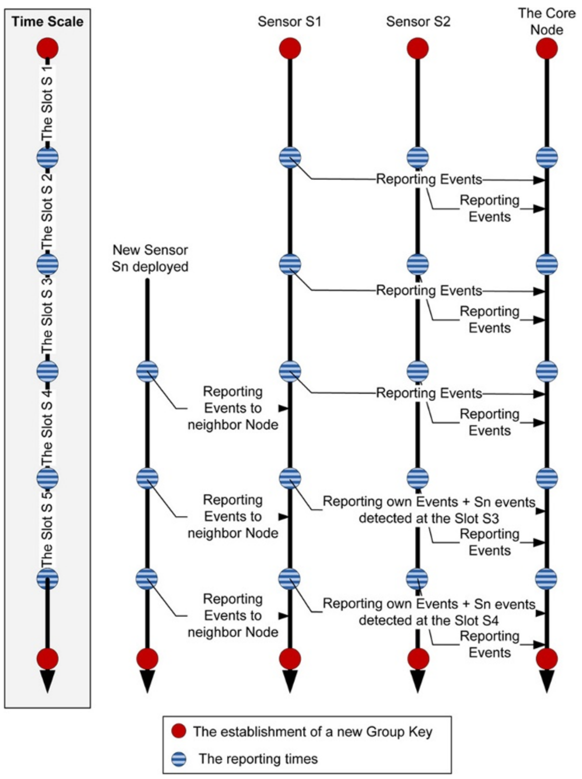
Time Scale
The Slot S 1
The Slot S 2
The Slot S 3
The Slot S 4
The Slot S 5
Sensor S1
Sensor S2
The Core Node
New Sensor Sn deployed
Reporting Events
Reporting Events
Reporting Events to neighbor Node
Reporting own Events + Sn events detected at the Slot S3
Reporting own Events + Sn events detected at the Slot S4
● The establishment of a new Group Key
● The reporting times

Figure 5 The integration of a new sensor.

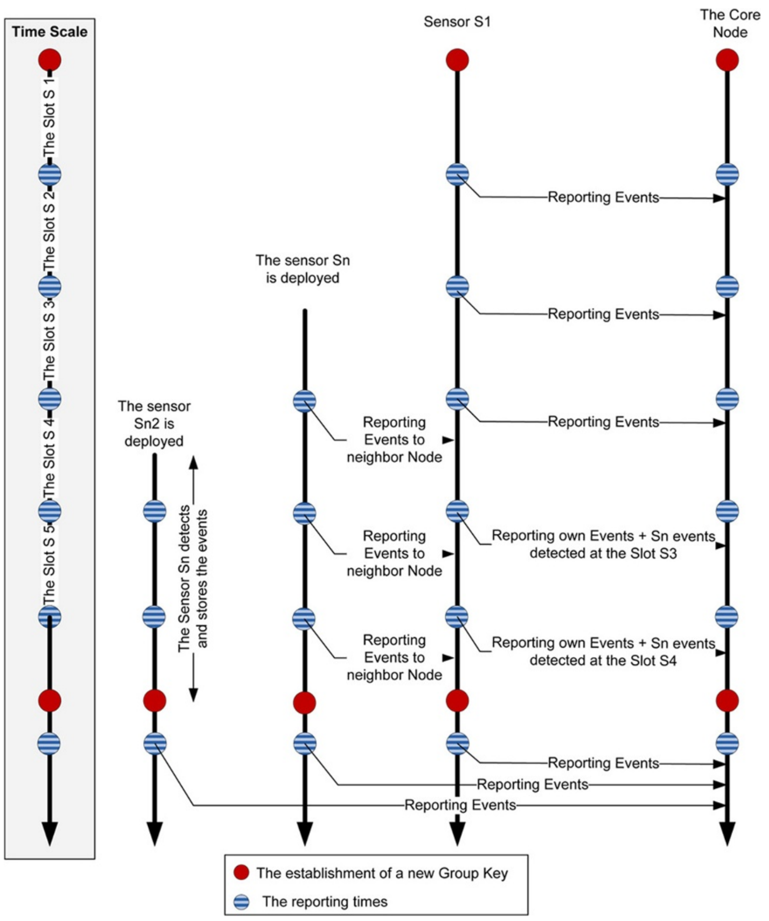
Time Scale
The Slot S 1
The Slot S 2
The Slot S 3
The Slot S 4
The Slot S 5
Sensor S1
The Core Node
The sensor Sn is deployed
The sensor Sn2 is deployed
The Sensor Sn detects and stores the events
Reporting Events
Reporting Events to neighbor Node
Reporting own Events + Sn events detected at the Slot S3
Reporting own Events + Sn events detected at the Slot S4
● The establishment of a new Group Key
● The reporting times

Figure 6 The time-dependent integration of a new sensor.

7 Robustness of the algorithm

In this part, we will prove the robustness of the proposed security protocol relative to the exchanged messages and the global security architecture and validity of the key.

7.1 Messages robustness

In this part, we will analyze the robustness of each one of the messages exchanged in the establishment of the group key. For each message, we will analyze its ability to support the authenticity, the confidentiality, and the integrity of the sent data:

- The message : {Type_*of*_message, [Identifier,g^{s*Ri}]}, $\{AUTH\}_{KSi}$
  This message is sent by each sensor to the core node to present its contribution in the group key:
  - The authentication is verified because the field AUTH is encrypted using the private key of the sensor KSi.
  - The integrity is done by the field AUTH which is a digest of the original message. This part cannot be modified by an intruder because it necessitates the use of the private key.
  - The original data is sent in clear mode, but the confidentiality is ensured. An intruder cannot use this information. The important information sent is the contribution of the sensor which is $\mathbf{g}^{s*\mathrm{Ri}}$. However, the values of *S* and *g* are secret values and are deployed in trusted nodes and are not known for the intruder. Then, it cannot separate the value $\mathbf{g}^{s*\mathrm{Ri}}$ into separate correct values of s and Ri.
- The message : {[Identifier of S1, $g^{S*R1*Rc}$],..., [Identifier of Si, $g^{S*Ri*Rc}$],..., [Identifier of Sn, $g^{S*Rn*Rc}$]}, $\{AUTH\}_{KCN}$
  This message is the message in which the core node sends all the contributions of the sensors to all the nodes. Those values will be used to calculate the group key:
  - The authentication is verified because the field AUTH is encrypted using the private key

of the core node KCN. Then, an intruder cannot sign the message using this key.
  - The integrity is done by the field AUTH which is a digest of the original message. This part cannot be modified by an intruder because it necessitates the use of the private key of the core node.
  - The original data is sent in clear mode, but the confidentiality is ensured. An intruder cannot use this information. The important information sent is the contributions of the sensors which are the sets of $\mathbf{g}^{S*\mathrm{Ri}*\mathrm{Rc}}$. However, the values of S and g are secret values and are deployed in trusted nodes and are not known for the intruder. Then, that one cannot separate the value $\mathbf{g}^{S*\mathrm{Ri}*\mathrm{Rc}}$ into separate correct values of S, Ri, and Rc

- The messages
  - $\{\mathrm{Mid, SAi}\}_{GKDH}\ \{\mathrm{digest}\}_{GKDH}$
  - $\{\text{list of sensors , cryptographic suite}\}_{GKDH}$, $\{\mathrm{digest}\}_{GKDH}$
  - $\{\mathrm{Mid,ID\_Sensor,SAr}\}_{GKDH}\}$, $\{\mathrm{digest}\}_{GKDH}$

  In those messages, the core node and the sensors establish the preferences for the CSA. Those three messages uses the same techniques to ensure the security requirements:

  - The authentication is verified because the field digest is encrypted using the group key. This key is only known by the trusted sensors.
  - The integrity is done by the field digest which is a digest of the original message. This part cannot be modified by an intruder because it necessitates the use of the group key.
  - The message sent is encrypted using the group key, and the intruder cannot decrypt the data because the group key has been distributed in a secure manner.

7.2 Key validity

In addition to the security performed in the key negotiation steps, we have another propriety of our protocol. This propriety gives a more robust security scheme. In fact, the established tunnels have no infinite validity. As presented previously in the specifications of the proposed protocol, the group key is periodically established with the contribution of the different communicating nodes that belong to the same group. From another point of view, a sensor that belongs to one group and has moved to another group will be automatically deleted from its initial group and affect the second group. This propriety enhances the global security of our protocol, because the sensor S1 that changes from one group G1 to another group G2 will not be able to decrypt the messages of the group G1. In fact, in the regeneration of the keys, the corresponding node S1 will share the key with the sensors that belongs to its newest group G2 and then the old key of the group G1 (known by S1) will no longer be a valid key and it cannot use it to decrypt the messages of its initial group. This is a protection against unauthorized access. In the case where the key does not change periodically, the sensor that has moved from one group to another can maintain a copy of its initial group key. Then, it can hear and decrypt the messages of the first group without belonging to this group. However, for our solution, this problem is automatically resolved because the keys are not of infinite validity and the sensor will have only a valid key used in the group to which it belongs in the current instant.

7.3 The effect of a node compromise

Until now, we have demonstrated the robustness of the CSA establishment process relative to the messages robustness and key validity. The network implemented operates in the general cases in a hostile environment and the sensor nodes can be compromised by two means.

For the first compromise method, the attacker will destroy the sensor to be in a denial of service. For the process of the group key establishment, if a node is not operating, the group key and the CSA will be correctly established. In fact, the list of the sensors that should collaborate to establish the group key is not predetermined, and any trusted nodes can participate in the establishment of the key in a secured manner. Then, if a node is destroyed, the proposed key establishment process is not affected.

The second type of a node compromise is a trial of the sensors content reading. The aim of this compromise is to use the node ID and the keys stored in the sensor node. If this is done, the intruder will be able to participate in the key establishment process. To prevent this kind of compromise, we propose two preventive mechanisms:

- For the first protection solution, the memory contents stored in the sensors should be encrypted. In that case, if an intruder tries to access the content, it should decrypt it. If we use a strong cryptographic scheme, the attacker will not be able to read in clear the ciphered content.
- For the second protection solution, we propose a physical protection of the nodes by the use of self-destructive sensors. This kind of sensors is automatically destructed at any attempt of an external sensor manipulation or attempt of content reading. In that case, if any intruder tries to read the content, the node will be automatically destructed

and then the memory that contains the cryptographic keys and security information is flushed.

By the mean of these two protection mechanisms, the intruder will not be able to use the content of the sensor either for sending data or participating in the group establishment process because the information stored in the sensor will not be available.

8 Simulations

In this section, we will present in the first part the simulation model used in the evaluation of DynTunKey. This model contains respectively the deployment model and the generation of the events used in the simulation. We will then represent the results of some conducted simulations. In the first simulations, we will compare the proposed protocol DynTunKey to some recent works that propose a key management solution for wireless sensor networks. The second simulations aim to compare our protocol to the classic IPSec protocol [21,22].

8.1 Simulation model

8.1.1 Deployment of the sensors

The sensors are deployed in the area to ensure a k-covered area. The deployment strategy is described in [23,24]. We divided the area into zones. As presented in the proposed protocol, the sensors that belong to the same zone establish a shared tunnel with the core node. We divided the lifetime of the sensors into slot times. As presented in the communication model, each sensor will report its collected data to the core node at the end of the slot time. The group key validity is a set of slot times.

8.1.2 Deployment of the targets

In conventional sensor networks, static targets are used and considered in the research works. Our study in this paper is interested by securing the detected events. In our case, we will consider many targets in the monitored region, and those targets are moving. Initially, we deployed random targets in the covered area of the wireless sensor network. Those targets are mobile. We considered respectively two models of mobility which are the Random Walk and Gauss Markov mobility models [25]. For each of those models, we evaluated some metrics to compare our proposed protocol and other key management solutions. The simulations and results will be presented in the next sections. In this subsection, we will introduce those two mobility models:

- *The Random Walk model*. Starting from the initial distribution of the targets at time t_0, we assume that each target performs a 2-D random walk movement. With this mobility model, each target T_i travels from its current location to a new location by randomly choosing a direction $\theta \in [0, 2 * \pi]$ and a distance $d_i \in [d_{\min}, d_{\max}]$ in a prefixed time interval Δ_t. For our case, each target will have its own distance and direction in each time interval. Considering the initial positions $X_i(t_0)$ and $Y_i(t_0)$ of a particular target at time t_0, the formulas 1 and 2 give the new position of a target T_i after the random movement.

$$X_i(t_0 + \Delta_t) = X_i(t_0) + d_i(t_0 + \Delta_t) * \cos(\theta_i(t_0 + \Delta_t)) \tag{1}$$

$$Y_i(t_0 + \Delta_t) = Y_i(t_0) + d_i(t_0 + \Delta_t) * \sin(\theta_i(t_0 + \Delta_t)), \tag{2}$$

where $d_i(t_0 + \Delta_t)$ and $\theta_i(t_0 + \Delta_t)$ are random variables and represents respectively the distance and direction of displacement that occurs in the period between t_0 and $t_0 + \Delta_t$.

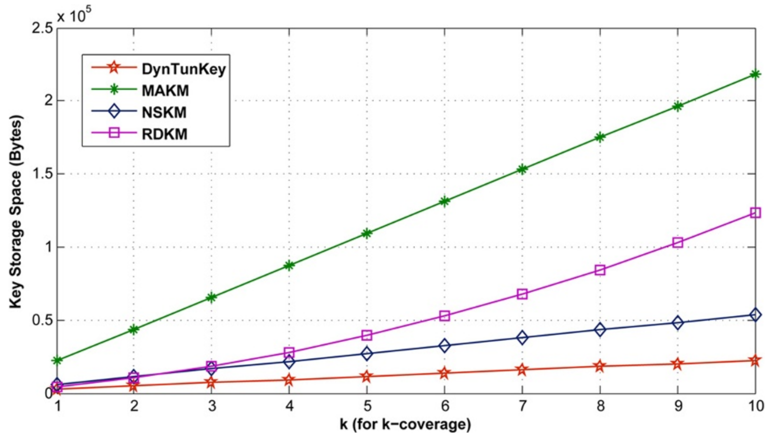

Figure 7 Key storage space for Random Walk.

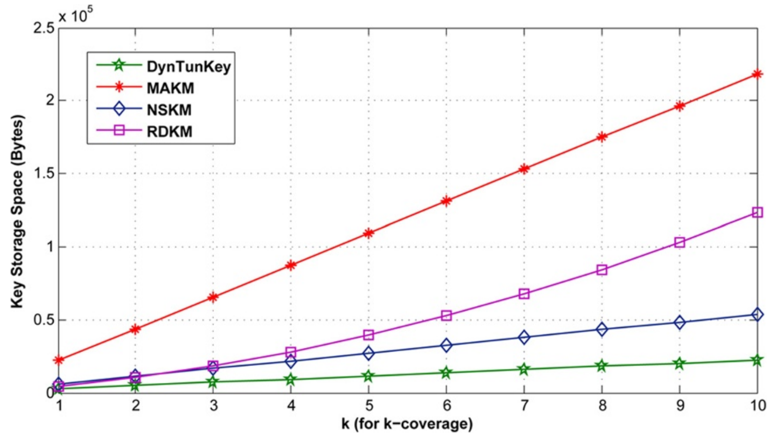

Figure 8 Key storage space for Gauss Markov.

This mobility is done in every time interval Δ_t and for every target T_i.

- *The Gauss Markov model*. The Random Walk model is a pure random model. In fact, for this model, there is no logic moving of the targets. The Gauss Markov model is considered to be a more realistic mobility model. In fact, it relates the current displacement distance $d_i(t_0 + \Delta_t)$ and the current direction $\theta_i(t_0 + \Delta_t)$ with its previous displacement $d_i(t_0)$ and direction $\theta_i(t_0)$. Thus, this model takes into account the previous movements of each target and then we will have a more realistic mobility path either in direction or speed.
  The formula 3 gives the new displacement $d_i(t_0 + \Delta_t)$ relatively to $d_i(t_0)$

$$d_i(t_0+\Delta_t) = a*d_i(t_0)+(1-a)*\bar{d}+\sqrt{1-a^2}*X_d, \tag{3}$$

where a is the tuning parameter $\in[0,1]$, $\bar{d}$ is the average displacement distance in the interval Δ_t relatively to the speed of the target, and X_d is a random variable of the Gaussian distribution.
The formula 4 gives the new direction $\theta_i(t_0 + \Delta_t)$ relatively to the previous direction $\theta_i(t_0)$

$$\theta_i(t_0 + \Delta_t) = \theta_i(t_0) + \sqrt{1-b^2} * X_\theta, \tag{4}$$

where b is the tuning parameter $\in[0,1]$ and X_θ is a random variable of the Gaussian distribution.
Then, the new positions are given using Equations 5 and 6.

$$X_i(t_0+\Delta_t) = X_i(t_0)+d_i(t_0+\Delta_t)*\cos(\theta_i(t_0+\Delta_t)) \tag{5}$$

$$Y_i(t_0+\Delta_t) = Y_i(t_0)+d_i(t_0+\Delta_t)*\sin(\theta_i(t_0+\Delta_t)). \tag{6}$$

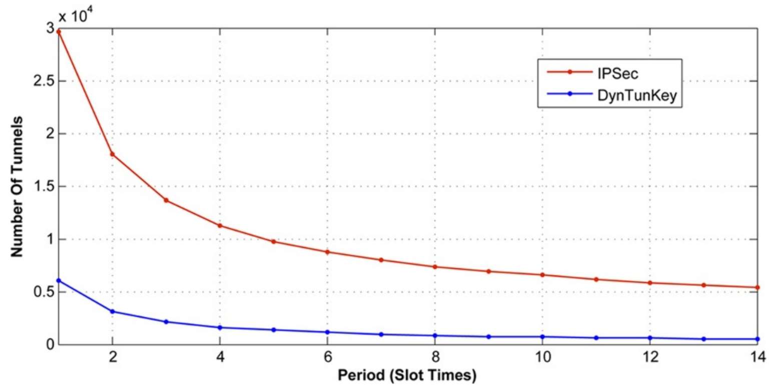

Figure 9 The number of tunnels for the Random Walk model.

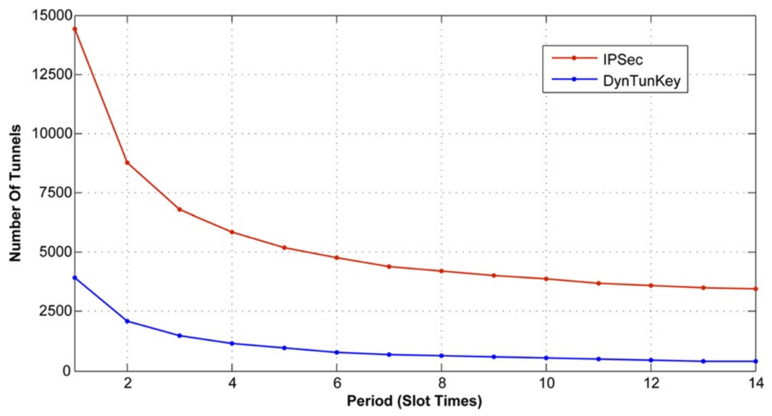

Figure 10 The number of tunnels for the Gauss Markov model.

8.2 Comparison of DynTunKey to MAKM, NSKM, and RDKM

In this first part of the simulations, we will compare our proposed protocol DynTunKey to recent key management protocols for wireless sensor networks. In particular, we will consider the protocols MAKM [26], NSKM [15], and RDKM [16]. We will compare those mechanisms by evaluating the key storage space required for the keys used in all the network. We considered many densities in the deployment of the sensors to ensure k-coverage. For each one of those values, we deployed targets in the covered area and evaluated the storage space required for the network keys. As used in the simulation of the previous works, the size of each key is 20 bytes. Figures 7 and 8 represents those variations considering respectively the Random Walk and Gauss Markov mobility models for the deployed targets.

The first deduction in those simulations is that our proposed protocol DynTunKey outperforms all the other solutions relative to the key storage space. The proposed protocol gives good performances either considering the Gauss Markov mobility or the Random Walk mobility. This result is logic, because for DynTunKey the number of the keys needed and stored in the nodes of the network does not depend only on the number of the sensors but depends directly on the occurrence of a target behavior in the monitored area. In fact, all the mentioned solutions establish keys for all the network nodes despite the reporting necessity. Then, as a consequence, the performances of DynTunKey will be better or at least will give the same performances of the other protocols when all the sensors establish keys with the core node.

Another remark is that when the sensor density is greater, the shift between the storage space for our solution and those of others becomes greater. For example, when considering the 1-coverage and 2-coverage densities, the key storage space for DynTunKey is closer to the values of the other protocols. However, when the value of k becomes greater, the key storage space is larger. In fact, as said previously, the number of created and stored keys in DynTunKey is dynamic and depends not only on the number of the sensors but also is established for only the sensors that have to report their detections. Then, the number of the keys is adapted to the number of events that occurs in the monitored area. However, for the other solutions, the number of the stored keys depends only on the number of the sensors and then the relation between the number of the sensors and the number of the keys is direct.

As a general conclusion, DynTunKey gives good performances when compared to the other protocols in all the sensor density cases and target behavior.

8.3 Comparison of DynTunKey and IPSec

In the previous section, we compared the proposed protocol DynTunKey to recent proposed key management protocols. Those protocols are used in wireless sensor networks but does not use the concept of tunneling. To the best of our knowledge, our solution is the first one that uses the concept of tunnels to ensure the security for the wireless sensor networks. Based on those facts, we will compare DynTunKey to another protocol based on the tunneling management.

We will represent the results of some conducted simulations. The aim of those simulations is to compare our protocol to the classic IPSec protocol [21,22]. The metrics that will be used in the comparison are respectively the number of the tunnels, the number of messages, and the security latency.

Table 2 The standard deviation of the number of tunnels

	Period ∈ [1,6]	Period ∈ [7,14]
The RW model	1,860	160
The GM model	1,167	106

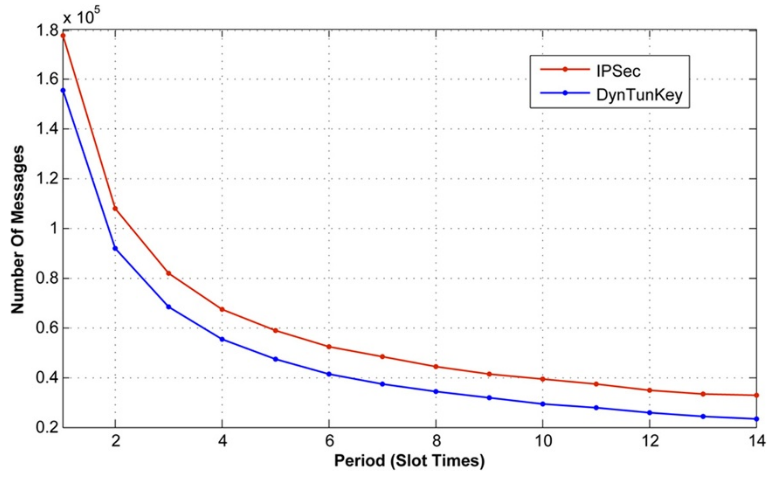

Figure 11 The number of messages for the Random Walk model.

8.3.1 Number of tunnels

For this first part of the simulation, we will compare DynTunKey and IPSec regarding the number of created tunnels. We considered different values of the key validity period going from 1 slot time to 14 slot times. For each of those values, we calculated the number of created tunnels for IPSec and the proposed protocol.

Figure 9 represents the variation of the created tunnels for each one of the protocols considering the Random Walk mobility model. Figure 10 represents the variation of the created tunnels for each one of the protocols considering the Gauss Markov mobility model.

Figures 9 and 10 show that the number of the created tunnels for the protocol is less than those created for IPSec. In fact, when considering the IPSec protocol, each event will be reported in a separate manner. Then, the number of tunnels is equal to the number of the detected and reported events. However, for our protocol, the events are gathered and reported at the end of each reporting period. Also, all the sensors that belong to the same zone establish a unique tunnel. Then, for DynTunKey, a set of sensors (belonging to the same zone) and a set of events (that occurs in the same period of key validity) share the same tunnel.

From those facts, it is logical that the number of tunnels for IPSec is greater than those for the proposed protocol DynTunKey. As a conclusion, this part of simulations shows that despite the model of mobility and behavior of targets, the proposed protocol gives a better performance than IPSec when considering the number of created tunnels.

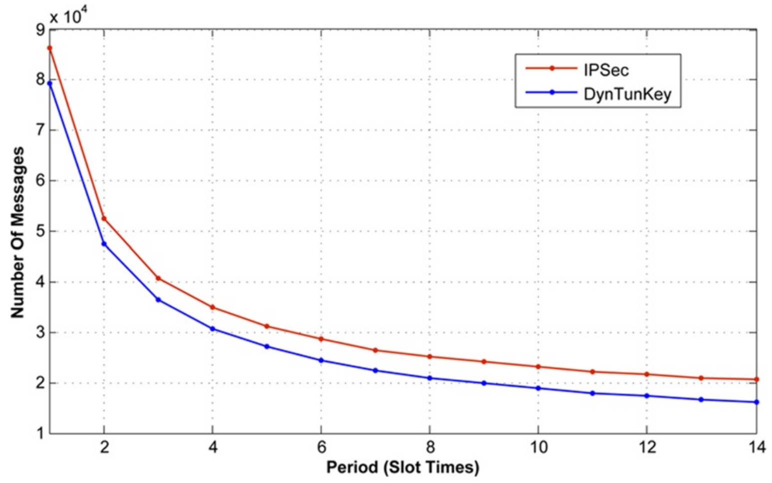

Figure 12 The number of messages for the Gauss Markov model.

Table 3 The standard deviation of the number of messages

	Period ∈ [1,6]	Period ∈ [7,14]
The RW model	42,745	4,948
The GM model	20,466	2,191

It comes also from those simulations that the number of tunnels decreases according to the period. In fact, we considered the same events with the same mobility, and when the value of period is greater, a more number of events are reported using the same shared group key. Thus, when the value of the period increases, the number of tunnels decreases.

However, we have to remember that a great value of key validity period means that the group key will be used for a longer time which is not suitable for the security robustness. Thus, we have to choose a median value of the period to ensure both a small number of tunnels and an accepted value of the group key periodicity regeneration.

Table 2 represents the standard deviation of the number of tunnels. We measured the standard deviation of the number of tunnels for the small values of the period and the big ones.

In this table, we can observe that for the two models of mobility, the variation of the number of tunnels is great when considering the small values of the period (ranging from 1 to 6 slots). However, for the big values of the period, the difference between two successive periods is not big when compared with variation between the small values of the period. In fact, when considering the smallest values of the period between 1 and 6 slots times, we will have a standard deviation equal to 1,860 for the Random Walk model and equal to 1,167 for the Gauss Markov model. However, for the biggest values of the period in the interval from 7 to 14 slot times, the standard deviation is equal to 160 for the Random Walk model and 106 for the Gauss Markov model.

Thus, from the experiments and results, we can determine the best value of the period. For the simulated case, we can select the value of the period in the first interval [1, 6] that minimizes the value of the number of tunnels when compared with the previous period. Also, the most adequate period is the one from which we will not have big variance with the tunnel number of the next periods. When considering this value, we can ensure two major proposed advantages of DynTunKey which are respectively the smallest value of the key period validity (to decrease the risk of an attack) and almost the minimal value of number of tunnels that can be reached which is one of the principal goals of DynTunKey.

8.3.2 Number of messages

For this simulation, the metric evaluated is the number of exchanged messages between the sensors and the core node. We considered different values of the period going from 1 slot time to 14 slot times. For each one of those values, we calculated the number of exchanged messages for IPSec and the proposed protocol. As well as the previous simulation, we considered two models of mobility which are the Random Walk and the Gauss Markov mobility. The variation of the number of messages in relation with the period are represented in Figures 11 and 12.

It shows in Figures 11 and 12 that the number of messages decreases according to the period. For the two mobility models, when comparing the proposed protocol and IPSec, the number of messages exchanged to establish all the tunnels for the proposed protocol is less than those exchanged for IPSec.

This is a logical result because for IPSec every event is reported at a separated manner and then messages are exchanged for every event detected to establish a dedicated tunnel. However, for the proposed protocol, many events detected by a sensor node are reported on a

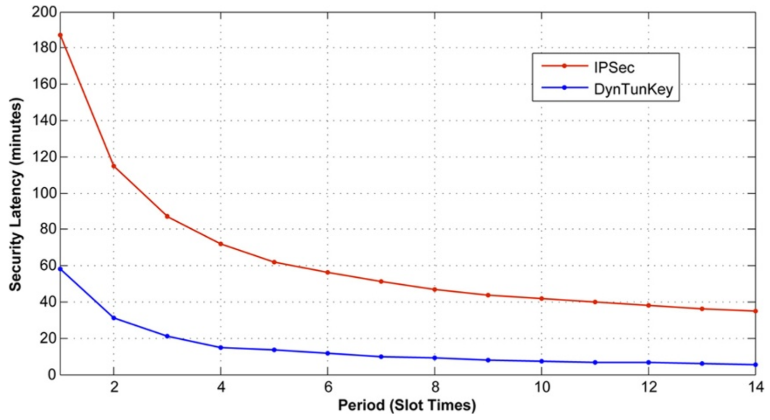

Figure 13 The latency time for the Random Walk model.

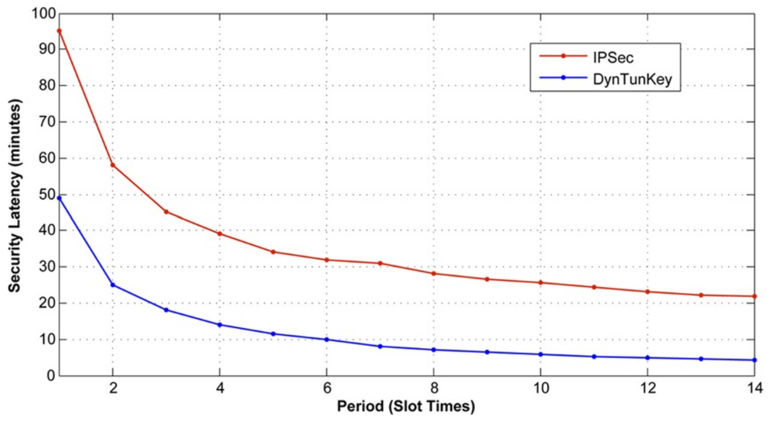

Figure 14 The latency time for the Gauss Markov model.

same tunnel. Thus, there is only one exchange of tunnel establishment messages for the proposed protocol. This establishment is performed at the beginning of the key validity interval. Hence, we can deduce that the proposed protocol outperforms IPSec when considering the communication overhead resulting from the establishment of the encrypted tunnels.

As well as for the number of established tunnels, we represent in Table 3 the standard deviation of the number of messages. We represent the standard deviation for small and big period values, when considering the two mobility models. The same conclusions can be deduced. For the small values of period, we have a big variance between the neighboring values, but for the big values of the periods, the deviation is less than the one for the small values. Then, the chosen value has to be in the small values of period. This value will verify the two criterions. At the first point, it is smaller enough to reduce the key validity interval and it gives also a remarkable variance of the number of messages when compared with the next values.

8.3.3 Security latency

In this third part of the simulation, we compare the proposed protocol and IPSec regarding the time required to establish all the tunnels. We developed two programs, the first one implements the exchange of messages as described in the proposed protocol. The second program implements the exchange of the messages to establish an IPSec tunnel. The same list of events is used for both protocols, and we considered such as the two previous simulations the Randow Walk and Gauss Markov mobility models. The two programs were executed on the same laptop, and we measured the total time of execution of each program. The results of this simulation are represented in Figures 13 and 14.

Despite the mobility model, the time required to establish the tunnels for the protocol is less than the time required for IPSec. This is a logical result because when considering IPSec, the tunnel is created in a separate manner because each sensor exchanges its specific messages with the core node. However, for the proposed protocol, only at the end of each period many sensors exchange simultaneously messages with the core node to establish a common tunnel or CSA. Also, the number of tunnels in the proposed protocol is less than those for IPSec and then the time of creation of tunnels will be certainly lower.

As well as the previous metrics, Table 4 represents the standard deviation of the latency time. For the same reasons, the best chosen period has to be in the small values. Then, a median value of the regeneration period will guarantee a small key validity interval and an optimal value of the number of tunnels, number of messages, and establishment latency when compared with the previous and next period values.

9 Conclusion

In this paper, we proposed DynTunKey which is a secure group key and tunneling management protocol for wireless sensor networks. This protocol aims to establish dynamic secure tunnels between the nodes. To optimize the creation of the tunnels, the protocol creates a shared cluster security association common to the sensor nodes that detect the same event and belongs to the same geographic zone. The proposed protocol has many advantages. Regarding the security, it permits a dynamic generation of a periodic group key. This shared group key is a symmetric key and then the encryption of the data uses less computational resources than an asymmetric solution. In addition to the inputs in terms of

Table 4 The standard deviation of the security latency

	Period ∈ [1,6]	Period ∈ [7,14]
The RW model	17.61	1.46
The GM model	14.6	1.45

security, the proposed protocol is useful in many applications that necessitates a secure communication among all the nodes such as military target tracking or firefighting collaborating team. To make our solution dynamic in an architectural term, we proposed a solution to integrate a newly deployed sensor in the secured set of the sensors. We conducted some simulations to evaluate the performance of the proposed protocol, and we have shown that the proposed approach considerably reduces the key storage space, the processing and communication overhead. In those simulations, we considered two mobility models of the targets to show that the model gives good results in all the cases.

Competing interests
The authors declare that they have no competing interests.

Received: 16 August 2012 Accepted: 7 January 2014
Published: 17 January 2014

References

1. M Horton, D Culler, K Pister, J Hill, R Szewczyk, A Woo, MICA, The commercialization of microsensor motes. Sensors. **19**(4), 40–48 (2002)
2. H Chan, A Perrig, D Song, Random key predistribution schemes for sensor networks. IEEE Symposium on Security and Privacy (IEEE, Piscataway, 2003)
3. W Du, J Deng, Y Han, P Varshney, A pairwise key pre-distribution scheme for wireless sensor networks, in *Proceedings of the Tenth ACM Conference on Computer and Communications Security (CCS 2003)* (Washington, DC, 27–30 October 2003), pp. 42–51
4. SA Cametepe, B Yener, Combinatorial design of key distribution mechanisms for wireless sensor networks, in *Proceedings of the 9th European Symposium Research Computer Security* (Sophia Antipolis, 13–15 September 2004)
5. L Eschenauer, VD Gligor, A key-management scheme for distributed sensor networks. Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS '02) (ACM Press, New York, 2002), 41–47
6. RD Pietro, Mancini LV, Mei A, Random key-assignment for secure wireless sensor networks. Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03) (ACM Press, New York, 2003), 62–71
7. Liu D, Ning P, *Establishing Pairwise Keys in Distributed Sensor Networks*, pp. 52–61
8. W Du, J Deng, YS Han, S Chen, PK Varshney, A key management scheme for wireless sensor networks using deployment knowledge. Proceedings of the IEEE INFOCOM, Hong Kong (IEEE, Piscataway, 2004), 586–97
9. W Bechkit, New key management schemes for resource constrained wireless sensor networks, in *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM,* Lucca, 20–24 June 2011
10. TD Subash, C Divya, Novel key pre-distribution scheme in wireless sensor network. International Conference on Emerging Trends in Electrical and Computer Technology, ICETECT (IEEE, Piscataway, 2011), 959–963
11. A Rasheed, R Mahapatra, Key predistribution schemes for establishing pairwise keys with a mobile sink in sensor networks. IEEE Transactions on Parallel and Distributed Systems (IEEE, Piscataway, 2011), 176–184
12. R Di Pietro, LV Mancini, YW Law, S Etalle, P Havinga, LKHW: a directed diffusion-based secure multicast scheme for wireless sensor networks. 32nd International Conference on Parallel Processing Workshops (ICPPW '03) (IEEE, Piscataway, 2003), 397–406
13. H Chan, A Perrig, PIKE: peer intermediaries for key establishment in sensor networks. IEEE Infocom (IEEE, Piscataway, 2005)
14. S Zhu, S Setia, S Jajodia, LEAP: efficient security mechanisms for large-scale distributed sensor networks. 10th ACM Conference on Computer and Communications Security (CCS'03) (ACM Press, New York, 2003), 62–72
15. IS Gawdan, CO Chow, T Zia, QI Sarhan, A novel secure key management module for hierarchical clustering wireless sensor networks. Proceedings of the IEEE International Conference on Computational Intelligence Modelling and Simulation, CIMSim 2011, art. no. 6076377 (IEEE, Piscataway, 2011), 312–316
16. Y Zhang, X Li, J Yang, Y Liu, N Xiong, AV Vasilakos, A real-time dynamic key management for hierarchical wireless multimedia sensor network. Multimedia Tools Appl. **67**(1), 97-117 (2013)
17. R Bellazreg, N Boudriga, M Hamdi, A dynamic distributed key tunneling protocol for heterogeneous wireless sensor networks. IEEE International Conference on Trust Security and Privacy in Computing and Communications TrustCom (IEEE, Piscataway, 2012)
18. M Sliti, M Hamdi, N Boudriga, A Helmy, An elliptic threshold signature framework for k-security in wireless sensor networks. The 15th IEEE International Conference on Electronics, Circuits, and Systems (IEEE, Piscataway, 2008)
19. D Augot, R Bhaskar, V Issarny, D Sacchetti, An efficient group key agreement protocol for ad hoc networks, in *IEEE Workshop on Trust, Security and Privacy in Ubiquitous Computing, WoWMoM 2005,* Taormina, 13–16 June 2005
20. W Diffie, ME Hellman, New directions in cryptography. IEEE Trans. Inf. Theory. **22**, 644–654 (1976)
21. S Kent, K Seo, Security architecture for the internet protocol, RFC 4301. IEEE International Conference on Electronics, Circuits, and Systems (IEEE, Piscataway, 2005)
22. C Kaufman, E Microsoft, The Internet Key Exchange (IKEv2) Protocol, RFC 4306 (2005). http://tools.ietf.org/search/rfc4306. Accessed June 2012
23. R Bellazreg, M Hamdi, Boudriga N, Coverage control and irregular radio propagation in wireless sensor networks. WCNC IEEE Wireless Communications and Networking Conference (IEEE, Piscataway, 2009)
24. M Hamdi, N Boudriga, MS Obaidat, WHOMoVeS: an optimized broadband sensor network for military vehicle tracking. Int. J. Commun. Syst. **21**, 277–300 (2008)
25. T Camp, J Boleng, V Davies, A survey of mobility models for ad hoc network research. Wireless Commun. Mobile Comput (WCMC), **2**(5), 483–502 (2002)
26. D Du, H Xiong, H Wang, An efficient key management scheme for wireless sensor networks. Int. J. Distributed Sensor Netw. **2012**, 11–14 (2012)

doi:10.1186/1687-1499-2014-9
Cite this article as: Bellazreg and Boudriga: **DynTunKey: a dynamic distributed group key tunneling management protocol for heterogeneous wireless sensor networks.** *EURASIP Journal on Wireless Communications and Networking* 2014 **2014**:9.