HRKT: A Hierarchical Route Key Tree based Group Key Management for Wireless Sensor Networks

Rong Jiang[1], Jun Luo[1] and Xiaoping Wang[1]
[1] School of Computer, National University of Defense Technology, Changsha, China
[e-mail: {jiangrong, junluo, xiaopingwang}@nudt.edu.cn]
*Corresponding author: Rong Jiang

Received June 13, 2013; revised July 10, 2013; accepted July 22, 2013; published August 30, 2013

Abstract

In wireless sensor networks (WSNs), energy efficiency is one of the most essential design considerations, since sensor nodes are resource constrained. Group communication can reduce WSNs communication overhead by sending a message to multiple nodes in one packet. In this paper, in order to simultaneously resolve the transmission security and scalability in WSNs group communications, we propose a hierarchical cluster-based secure and scalable group key management scheme, called HRKT, based on logic key tree and route key tree structure. The HRKT scheme divides the group key into cluster head key and cluster key. The cluster head generates a route key tree according to the route topology of the cluster. This hierarchical key structure facilitates local secure communications taking advantage of the fact that the nodes at a contiguous place usually communicate with each other more frequently. In HRKT scheme, the key updates are confined in a cluster, so the cost of the key updates is reduced efficiently, especially in the case of massive membership changes. The security analysis shows that the HRKT scheme meets the requirements of group communication. In addition, performance simulation results also demonstrate its efficiency in terms of low storage and flexibility when membership changes massively.

Keywords: wireless sensor networks, group key management, route key tree, network security

A preliminary version of this paper was accepted as an invited paper in IEEE/CIC ICCC 2013, August 12-14, Xi'an, China. This version includes a concrete analysis and detailed security proof on key management. This research is supported by by China Scholarship Council, and the National Natural Science Foundation of China under Grant No.61170261.

http://dx.doi.org/10.3837/tiis.2013.08.017

1. Introduction

Wireless sensor networks (WSNs), which consist of spatially distributed autonomous sensors to monitor physical or environmental conditions, have attracted great attention from the industry and academia. Wireless sensor networks are fundamental parts of the internet of things (IOT) and have been widely used in many fields, such as battlefield surveillance, pollution monitoring, medical care and traffic control [1][2][3][4]. A wireless sensor network is usually composed of one or several base stations and a certain quantity of sensor devices, as shown in **Fig. 1**. The base station is used to collect data sensed from the sensor devices and transmits the aggregate data to the user. Sensor devices are basic components of wireless sensor network and the number of sensors in a wireless sensor network is from several hundreds to thousands or even more. The size of a sensor is small, and the sensor is usually equipped with a lithium battery, a microprocessor and a low memory. The sensor nodes communicate with each other by wireless channel and form a connective network in ad hoc mode.

Therefore, because of open wireless channels and constrained battery, memory and processor [5], wireless sensor networks are vulnerable to various kinds of attacks. It is of great importance to ensure secure communication in wireless sensor networks, especially when wireless sensor networks are deployed in hostile environments [6]. The messages transmitted in the network must be encrypted. Key management is one of the fundamental security mechanism to guarantee the communication security in wireless sensor networks [7][8]. The other security mechanisms such as secure routing, secure localization, authenticity and integrity are built upon the secure key management [9][10]. Group key [11][12] is one of the most important key management paradigms for group communication, which is both bandwidth-efficient and energy-efficient. Hence, establishing a secure and efficient group key for the resource-constrained wireless sensor networks is an important concern to inhibit an adversary from attacking group communication.

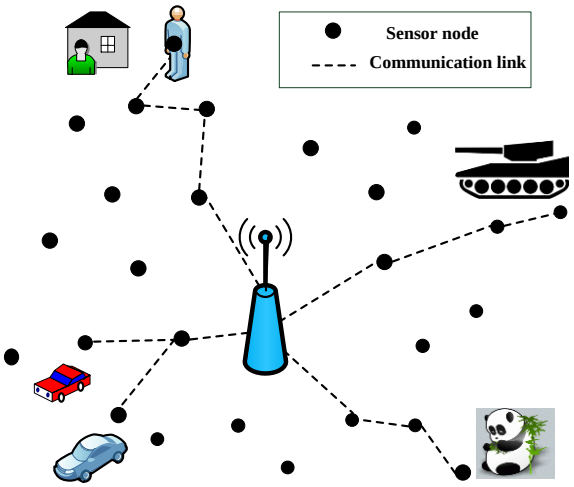

Fig. 1. The application areas of WSNs

Group key management is a challenging problem that has been considered as a vital issue in wireless sensor networks. Several group key management schemes for the wireless sensor

networks have been proposed. The logical key tree-based scheme, which can reduce rekeying cost dramatically by constructing a tree of key encryption keys, is one of the significant schemes.

The Logical Key Hierarchy (LKH) [13] scheme is a secure and scalable group rekeying technique which is designed for conventional wired networks at the beginning. By means of constructing a tree of key encryption keys, it can reduce the number of rekeying messages to $O(\log N)$. Unfortunately, this LKH scheme can introduce huge communication cost whenever a node joins or leaves the multicast tree in multi-hop wireless sensor networks since the constructed key tree may be different from the wireless sensor network route topology. As a result, it is unsuitable for the wireless sensor networks.

Lazos et al. [14] and Di Pietro et al. [15] extend this scheme for multicast key distribution to wireless sensor networks. As a kind of centralized group key management scheme, they are often not ideal for wireless sensor networks. Huang [16] proposes the scheme SLIMCAST, which introduces a level structure to manage the keys and reduce the overhead of nodes joining and/or leaving groups. In the SLIMCAST scheme, the network is subdivided into levels and branches. There are different level keys used to decrypting or encrypting group messages in different levels of a branch. The rekeying cost can be reduced because only local level key has to be updated while a node joins or leaves the group. However, when the number of group member changes frequently, the performance will be degraded largely. TKH [17] scheme, which takes the advantage of subtree-based key tree separation and wireless multicast, constructs a key tree according to the structured sensor network route topology. The nodes which are topologically adjacent are assigned the same key encryption keys (KEKs) so that the total rekeying cost can be reduced. However, there are some potential safety hazards in this scheme and an adversary may get the keys during the key update.

In order to solve the transmission security and scalability in wireless sensor network group communications problems mentioned above, in this paper, we propose a Hierarchical Route Key Tree based group key management scheme, named HRKT, to solve this problem. In summary, the HRKT scheme is distinguished by the following features:

- Considering different security communication requirements among different types of sensor nodes, the proposed group key management is divided into two parts: cluster head key management and cluster member key management. The security level of communication among cluster heads is enhanced by hop-by-hop encryption.
- The key tree is generated according to the route topology of the wireless sensor network so that the rekeying messages are only sent to corresponding subtree and the communication cost can be reduced greatly.
- The hierarchical key structure facilitates local communication based on the fact that the nodes at a contiguous place communicate with each other more frequently. Different keys can be used in different ranges of communication. Even if some keys may be compromised, the other keys will not be affected.
- Because of the hierarchical architecture, the key update is confined in a cluster or a level. The key update messages can be combined together and transmit to the nodes which share the same key encryption keys. When membership changes are large-scale, the performance is efficient.
- A lazy update strategy is further leveraged to decrease the rekeying cost with acceptable delay.

The remainder of the paper is organized as follows. In section 2, we briefly introduce the system model, security model and design goal. HRKT scheme is proposed in Section 3, in which we introduce how to generate and update the key tree. We give a detail security and

efficiency analysis in section 4 and section 5, respectively. Finally, we draw our conclusion and future work in section 6.

2. Models and Design Goals

2.1 System Model

We use the notations listed in **Table 1** throughout the rest of this paper.

Table 1. Notations in HRKT scheme

Notation	Description
BS	Base Station
CH_i	Cluster Head *i*
PCH_i	the Parent node of CH_i
CM	Cluster Member node
ID	the IDentity of a sensor node
$H(\cdot)$	the one way hash function
ROF	one way sequence number computed by a one way hash chain
$M_1 \mid\mid M_2$	the concatenation of message M_1 and M_2
$E(m, k)$	encrypt the message m with key k
LK	Level Key shared by the CHs
PK	a node's Private Key shared with the base station
SK	a node's Secret Key shared with the CH
CK	Cluster Key shared by the nodes which are in the same cluster
BK	Branch Key shared by the nodes which are in the same branch tree
FK	Fellow Key shared by the nodes which have the same parent in a branch tree
e_{tx}	energy dissipated during 1-*bit* transmission by a sensor node
e_{rx}	energy dissipated during 1-*bit* reception by a sensor node

We list a set of assumptions concerning the topology and shared security inherent in the wireless sensor network used in the proposed scheme:

- There is a base station in the wireless sensor network, and it owns much more storage, communication and computation resources than the other nodes in the network.
- The sensor devices are static and homogeneous. We do not consider mobility scenarios in this paper.
- The network can dynamically group sensor nodes into clusters. There is a leader node referred to as Cluster Head (CH) in each cluster. Cluster heads are elected according to particular rules, and they can constitute a connected network. The cluster members can only communicate with their neighbors of the same cluster and transmit messages to the cluster head by one hop or multi-hops.
- Each node stores a private key (PK) shared with the base station and an initial one-way hash chain value, denoted as ROF , which is one way sequence number computed by a one way hash chain [18] when it is deployed. The key PK is used to securely communicate with the base station, and ROF is used for verifying the legitimacy of messages broadcast by the base station.

- With the help of base station, each cluster member node generates a Secret Key (SK) shared with the CH .

2.2 Security Model

Generally speaking, the security properties required by wireless sensor networks are shown as follows [19]: confidentiality, integrity, authentication, non-repudiation, authorization, and so on. Specifically, the following security requirements on group communication should be desired (Let J_j be a set of nodes newly joining the group; Let R_j be a set of revoked nodes):

- **Group confidentiality**: nodes which are not members of the group can not obtain any keys used in group communication.
- **Backward secrecy**: given any set J_j, it is computationally infeasible for the nodes $u_i \in \mathrm{J}_j$ to recover any former group keys $GK_1, \cdots, GK_{j-1}$ by sharing information, i.e., newly joining node should not be able to know any previous keys so that it cannot decrypt previously transmitted message before it joins the group.
- **Forward secrecy**: given any set R_j, it is computationally infeasible for the nodes $u_i \in \mathrm{R}_j$ to recover any of subsequent group keys $GK_j, \cdots, GK_m$ by sharing information, i.e., after a node has been removed from the group, it is not able to obtain any new keys.

2.3 Design Goal

Considering the constrained condition in wireless sensor networks, we try to propose an efficient group key distribution scheme to achieve the above security objectives. In particular, we will achieve:

- **Scalability**: The proposed HRKT scheme should achieve scalability in the wireless sensor networks. When the size of the wireless sensor network grows, it is difficult to maintain the same security level. Different key management policies may incur different sizes of wireless sensor network. Thus, how to determine the maximum supported network scale for a given key management policy is important. A scalable group key distribution scheme can support different scales of networks [20].
- **Efficiency**: The proposed HRKT scheme should also minimize the computation, memory, communication and energy costs in group communication. It should take into account sensor limitations, since the sensors such as the Mica2, which run industry standard protocols on 16 bit microprocessors with 4 kilobytes of RAM (working memory), and 128 kilobytes of FLASH (persistent memory), are resource constrained [21].

3. The Proposed HRKT Scheme

In this section, we describe the details of the proposed HRKT scheme, which consists of four parts: ***secure setup of the key tree, secure data multicast, node joining*** and ***node leaving***. According to the assumption before, the nodes in a wireless sensor network have dynamically grouped into clusters. There is a leader node referred to as Cluster Head (CH) in each cluster. CHs are elected according to particular rules, and they can constitute a connected network [22][23]. **Fig. 2** (a) illustrates an example of a sensor network topology.

We divide the keys into cluster head keys (i.e. inter-cluster keys) and cluster member keys (i.e. intra-cluster keys). In a group communication, the messages encrypted with cluster head key, which is referred to as level key LK in our scheme, are broadcasted to *CHs* first. For example, **Fig. 2** (b) illustrates a CH key tree structure based on the network topology of **Fig. 2** (a). The BS shares level key LK_1 with its child nodes CH_1 and CH_2. CH_1 shares LK_2 with its child nodes CH_3 and CH_4.

After verified and decrypted the messages, the CH encrypts the messages with cluster member key, which is referred to as cluster key CK in our scheme, and broadcasts them to its cluster member.

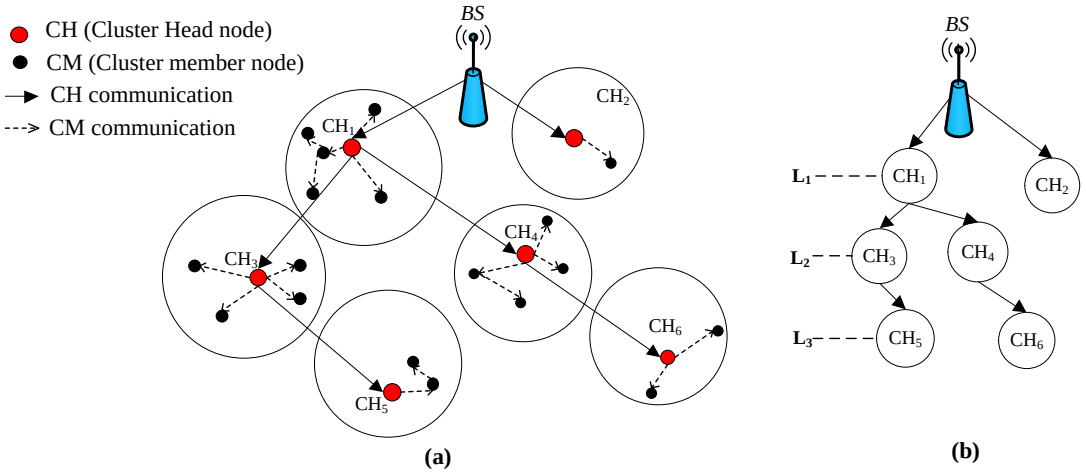

Fig. 2. (a) An example of a WSN topology; (b) CH key tree structure based on the network topology

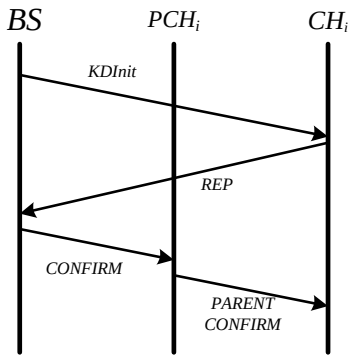

Fig. 3. Message sequence diagram of CH key tree generation

3.1 Secure Setup of the Key Tree

1) **Cluster Head Key Tree Generation**: Because *CHs* have to forward more messages collected from the other sensor nodes in the cluster, it is reasonable to enhance the communication security among cluster heads. We adopt and modify the idea of SLIMCAST for CH key tree generation to enhance the communication security. After construction of the cluster topology, the BS begins to set up the CH key tree. There are four phases to construct a CH key tree, as shown in **Fig. 3** and described below.

Phase 1: Initial Broadcast

The BS broadcasts an initial message *KDInit* to the *CHs*.After receiving this *KDInit* message, *CH* can respond a *Rep* message to join the multicast group. The format of the message *KDInit* is

$$BS \rightarrow CHs : \{KDInit \,||\, ROF_i \,||\, LastHop\}$$

where *KDInit* identifies the message as an initial message. The *ROF* field is one way sequence number computed by a one way hash function $H(\cdot)$ [18]. The sensor nodes will be pre-programmed with $H(\cdot)$ before they are deployed.

If the value of $H(ROF_i)$ computed form the *KDInit* message is equal to ROF_{i-1} stored in the *CH*, this *KDInit* is a legal message. Otherwise the *CH* will drop it. Therefore, the forged *KDInit* messages cannot be broadcasted in the network. The *ROFs* can prevent a replay attack.

The *LastHop* field is used to record the route information. The current *CH* stores the *ID* in the *LastHop* field as its parent and then rewrites *LastHop* field with its *ID*. This *LastHop* field information can be used to form a route for unicasting messages from a *CH* to the BS as well.

Phase 2: Join Reply

When the *KDInit* message is received, the *CHs* reply a message *Rep* to join the multicast tree. The format of the message *Rep* is

$$CH_i \rightarrow BS : \{Rep \,||\, ID_{CH_i} \,||\, E(TK_i, PK_{CH_i}) \,||\, MAC(\cdots, PK_{CH_i})\}$$

where *Rep* indicates the message as a reply to the BS. The message contains a new random key value TK_i encrypted with private key PK_{CH_i} which is shared with the BS. This TK_i is a session key which will be used to encrypt messages between the node and its parent. A $MAC()$ denotes the Message Authentication Code of the message and is encrypted using PK_{CH_i}. The suspension points $(\cdots)$ in $MAC()$ mean the content before $MAC()$, i.e., $Rep \,||\, ID_{CH_i} \,||\, E(TK_i, PK_{CH_i})$. Meanwhile, *CHs* generate a sliding window to store the average number of *Rep* messages which they have received within some time window. In case that this number exceeds a threshold, then *CH* will not forward any *Rep* messages until the average number of sliding window falls well below the threshold.

Phase 3: Confirm

After the BS receives and validates all *Rep* messages from the *CHs*, it will construct a *Confirm* message for each *CH*. The BS decrypts the TK_i field from the *Rep* message of CH_i and then appends it to the *Confirm* message of CH_i's parent node. Then the *Confirm* message will be unicasted to all the *CHs* one by one. The form of the message *Confirm* is

$$BS \rightarrow PCH_i : \{Confirm \,||\, ID_{PCH_i} \,||\, E(TK_i, PK_{PCH_i}) \,||\, MAC(\cdots, PK_{PCH_i})\}$$

Phase 4: CH Key Generation

Having received, verified and decrypted the *Confirm* message, each node which has child nodes will generate a level key LK_i for its child nodes. Then It will unicast a *ParentConfirm* message to each child CH_i. The form of the message *ParentConfirm* is

$$PCH_i \rightarrow CH_i : \{ParentConfirm \,||\, ID_{CH_i} \,||\, E(LK_i, TK_i)\}$$

The CH key tree generation procedure is finished.

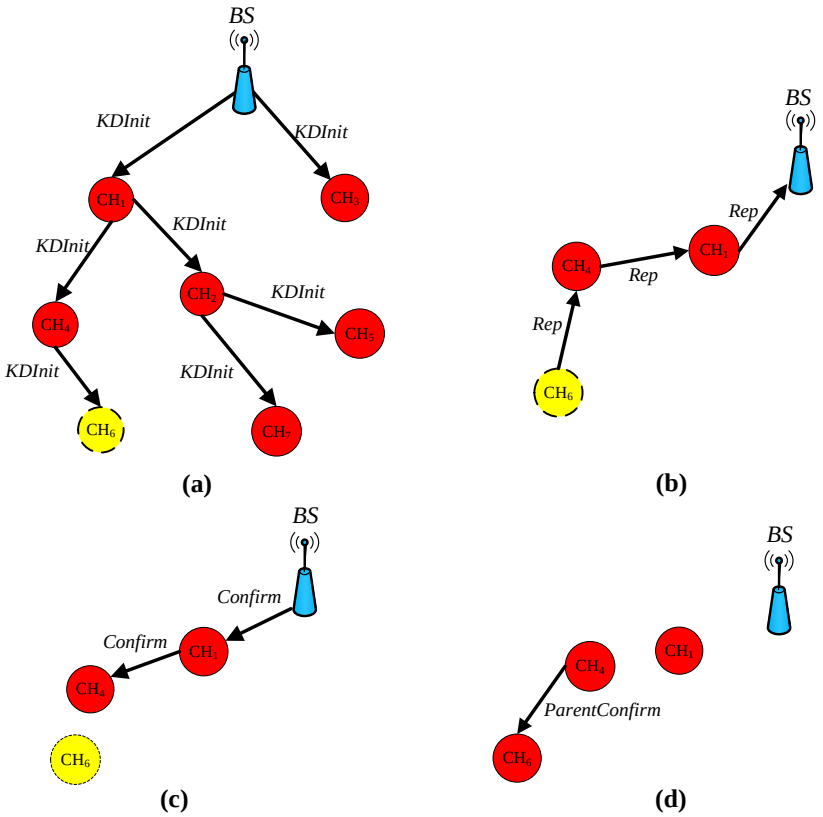

Fig. 4. An example of Cluster Head Key Tree Generation (a) The BS initializes CH key tree broadcasting; (b) CH_6 responses a Rep message to the BS ; (c) The BS constructs and send $Confirm$ message to the parent node of CH_6 ; (d) CH_6 receives level key from its parent node

Taking CH_6 in **Fig. 4** for example, after receiving and verifying the $KDInit$ message, CH_6 reply a message Rep ($\{Rep \,||\, ID_{CH_6} \,||\, E(TK_6, PK_{CH_6}) \,||\, MAC(\cdots, PK_6)\}$) to the BS . The BS decrypts this Rep message with PK_{CH_6} and obtains TK_6 . And then, after the BS recieves all the Rep messages from the CHs , the BS constructs a $Confirm$ message $\{ID_{CH_4} \,||\, E(TK_6, PK_{CH_4}) \,||\, MAC(\cdots, PK_{CH_4})\}$ encrypted with PK_{CH_4} and sends it to CH_4 (the parent node of CH_6). If CH_4 does not have a child node, it will generate a level key LK_3 for its child node CH_6 . Otherwise, it will send the level key that has been generated. Based on the $Confirm$ message, CH_4 constructs a $ParentConfirm$ $\{ID_{CH_6} \,||\, E(LK_3, TK_6)\}$ encrypted with TK_6 and sends it to CH_6 . CH_6 can obtain level key by decrypting this $ParentConfirm$ message. The other child nodes of CH_4 can proceed the same procedure to get the level key. After that, CH_4 can communicate with its child nodes by LK_3 securely.

2) **Cluster Member Key Tree Generation**: As we have mentioned before, the sensor

network have dynamically grouped into clusters. Once the new *CH* is securely elected by some particular rules, it begins to construct a cluster member key tree. The *CH* constructs a route tree in the first place. Based on the route tree, the *CH* assigns the same key encryption keys (KEKs) for geographically adjacent nodes. The advantage is that the nodes in the adjacent place can update the group key by one rekeying message.

First of all, the *CH* broadcasts a *RouteSetup* message to make cluster members set up paths to the *CH*. The format of the message *RouteSetup* is

$$CH \rightarrow CM : \{RouteSetup \,||\, ID_{CH} \,||\, HopCount\}$$

where *RouteSetup* field denotes this message as a route setup message; ID_{CH} indicates that this message is initiated by the *CH*, since only the cluster member nodes know the *ID* of the cluster head after secure *CH* election; The *HopCount* field stores the hop number to the *CH*, which will be increased by one before transmitted to the next node.

After receiving the *RouteSetup* message, a node compares the number of hops to the *CH* and picks the smallest one as its parent node (if there are several parents with the same hop number, the one with the highest SNR (Signal-to-Noise Ratio) will be chosen). Each member node broadcasts the *RouteSetup* message to its neighbors until the message reaches all nodes in the cluster. After that, each member reports its route information to the *CH* through a reverse path so that the *CH* can learn the whole cluster topology and each parent can learn and store its children's *IDs*. **Fig. 5** (a) is an example of a cluster member route tree.

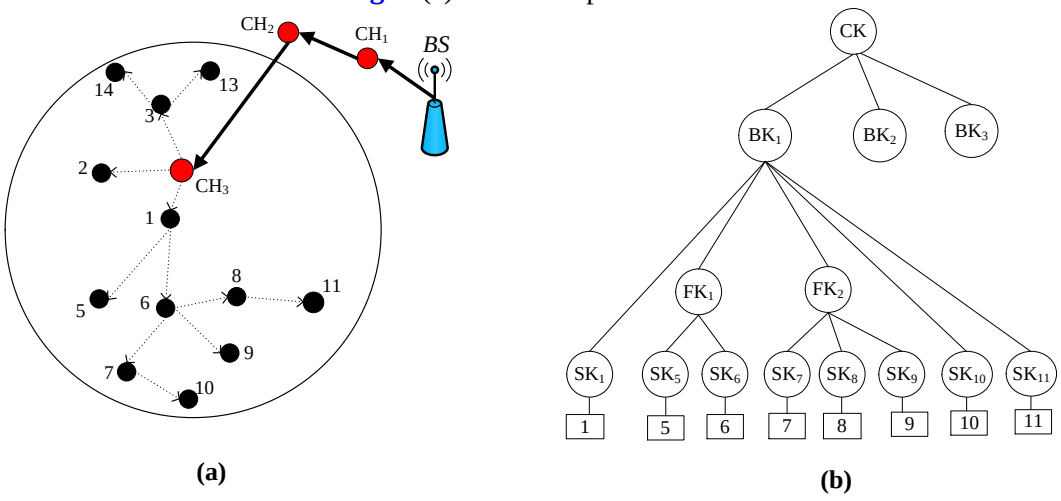

Fig. 5. (a) An example of a cluster member route tree; (b) the cluster member key tree structure based on the route topology

Based on the route information, the *CH* can build a route-based key tree. No matter what the size of a cluster is, we confine the depth of the key tree to 4 so that the storage overhead of keys is low in our scheme. These keys are Cluster Key (*CK*), Branch Key (*BK*), Fellow key (*FK*) and Secret Key (*SK*). Before a node joins the cluster, it must be verified by sending its *ID* and a *MAC* signature to the BS. After that, with the help of the BS, the node generates a secret key *SK* shared with the *CH*. The whole cluster nodes share a *CK* for cluster communication. The *CH* divides its direct child nodes into branches. Each branch node is the root of a branch tree and *BK* is shared by nodes in the same branch tree. If node membership changes in one of the branches, the cluster key can be updated by one message encrypted with

the corresponding BK in the other branches. The nodes, which have the same parent in a branch tree, share an FK. If a node in the branch tree has no sibling nodes, FK will not be generated and the node will just have CK, BK and SK.

For instance, **Fig. 5** (b) illustrates the cluster member key tree structure based on the route topology of **Fig. 5** (a). *Nodes* 1, 2, and 3 are the direct neighbors of the CH so that they are the branch nodes. *Nods*1and *nodes* 5-11 that are in the same branch share the BK_1. *Nodes* 5 and 6 which have the same parent share the FK_1. The node 10, which has no sibling nodes, only has BK_1, SK_{10} and CK.

After the key tree is constructed, the CH begins to distribute keys to each cluster member. The CH applies to the BS for a *seed* which is used to generate keys. With this *seed*, the CH computes corresponding keys in the finite field as follows:

$$CK = (g^{seed})^{R_{CK}},\ BK_i = (g^{seed})^{R_{BK_i}},\ FK_i = (g^{seed})^{R_{FK_i}}$$

where g is the generating element of a cyclic multiplicative group Z_p^*. Z_p^* is a cyclic multiplicative group of order p for some large prime p, R_{CK}, R_{BK_i} and R_{FK_i} are random numbers generated by the CH. And then, the CH unicasts corresponding CK, BK and FK encrypted with the corresponding SK to each node. After that, the CH will delete the seed and all the BK and FK. Even if the CH is compromised, the attacker cannot get all the keys. If the cluster needs to update the keys, the CH fetches the old seed and applies to the BS for a new seed. With the seeds, the CH can compute the old keys and generate new keys as before and update the corresponding keys. The CH will delete all seeds, keys and old random numbers after that. Although it may introduce some delay, the time of the update is acceptable since these seeds can be transmitted in one message. As a result, it mitigates the effect of compromised node.

3.2 Secure Data Multicasting

Because *CHs* have to gather all data in a cluster and forward them to the BS, communications among *CHs* are more important. Hence, it is reasonable to enhance the communication security among *CHs* and BS. Instead of using the group key, we use hop-by-hop encryption to multicast data among *CHs* and BS. In hop-by-hop encryption, the BS encrypts the message with its level key at the beginning. And then the BS broadcasts the message to *CHs* which are direct children of it. After receiving the message, the CH decrypts this message and re-encrypt this message using the level key shared with its child nodes. After that, the CH broadcasts this re-encrypted message to its child nodes. This hop-by-hop encryption will repeat until all the *CHs* receive the message. As shown in **Fig. 2** (a), the BS first multicasts a message to CH_1 and CH_3. The CH_1 decrypts the message with LK_1. Then the CH_1 re-encrypts the message with LK_2 and broadcast it to its child nodes. This procedure will repeat until the message reaches all the *CHs*.

In a cluster, the CH encrypts the message with the cluster key CK, and broadcasts it to all the cluster members. The one without CK cannot decrypt the message. Thus, the message is sent to each node securely.

One of the advantages of the proposed HRKT scheme is that the hierarchical key structure facilitates local secure communications of sensor nodes. The user of a wireless sensor network

usually interests in the data at a particular region. The scope of the region may be different, since different kinds of users have diverse interests. Therefore, different kinds of keys can be used in different scope of communications to make sure that the data are accessed by the minimum nodes. For example, in **Fig. 5**, the branch key BK_1 will be used to encrypt the messages if the user only interests in the area of *branch* 1. Even if the nodes in the area of *branch* 2 or *branch* 3 are compromised, the data transmitted in the area of *branch* 1 are still safe.

Furthermore, the proposed HRKT scheme can support source privacy preservation schemes [24][25] if necessary. The source location becomes extremely important especially when wireless sensor networks are deployed to monitor endangered species of animals or vital military targets, such as pandas or tank movements. An adversary with the knowledge of the location of the data source or base station may be able to infer the content of the data being transmitted or destroy the monitoring objects [5]. In order to protect the source location privacy, the CH can periodically collect real data's ciphertext from the cluster nodes if sensor nodes have detected some information, or dummy data's ciphertext otherwise. After receiving ciphertexts from sensor nodes, the CH will filter the dummy data, re-encrypt and forward the real data's ciphertexts to its parent node to achieve the source privacy preservation.

3.3 Node Joining

As soon as a sensor node newly joins the group or a new CH is elected in a cluster, in order to guarantee backward secrecy, the key tree should be updated. Two kinds of node joining events are described as follows.

1) *CH* Joining: When some sensor nodes are newly deployed or the old CH is not suitable to be a cluster head for some reason any more, a new CH need to be elected in the cluster. As soon as the new CH is elected, it will query its neighbour *CHs* about their hop count to the BS. After receiving all the messages from its neighbour *CHs*, the new CH sends a *Join* message to BS. The format of the message *Join* is

$$CH_i \rightarrow \mathrm{BS}: \{Join \,||\, ID_{CH_i} \,||\, ID\ of\ Neighbour \,||\, E(TK, IK_{CH_i}) \,||\, MAC(\cdots, IK_{CH_i})\}$$

where *Join* field denotes this message as a CH joining message; *ID of Neighbour* field contains all neighbour *ID* of the new CH. After verified this *Join* message, the BS will select a proper CH as its parent node and send a *JoinRep* message to its parent node PCH_i. The format of the message *JoinRep* is

$$\mathrm{BS} \rightarrow PCH_i: \{JoinRep \,||\, ID_{PCH_i} \,||\, E(TK_i, PK_{PCH_i}) \,||\, MAC(\cdots, PK_{PCH_i})\}$$

When the parent node PCH_i receives this *JoinRep* message, it will generate a new level key LK_i and construct a *ParentConfirm* message to the new CH_i. The form of the message *ParentConfirm* is

$$PCH_i \rightarrow CH_i: \{ParentConfirm \,||\, ID_{CH_i} \,||\, ROF \,||\, E(LK_i, TK_i)\}$$

The new CH can verify future *KDInit* messages by computing the *ROF* value. In order to guarantee backward secrecy, the level keys and cluster key need to be updated. The previous child node of the parent node PCH_i will receive the new level key LK_i encrypted with the previous level key LK_i'. After that, the CH key bootstrapping process will carry out as shown in section 3.1.

2) Cluster Member Node Joining: When a sensor node is newly deployed or fails to connect to its parent node, it needs to take part in a cluster. The new node sends a *JoinRequest* message with its *ID* and *MAC* to *CH*. The *CH* will send this message to BS. After the BS verifies the legitimacy of the joining node, with the help of BS, the node generates an *SK* shared with the *CH*. The *CH* then picks a node with the smallest hop number to the *CH* as its parent node based on the route tree. In order to guarantee backward secrecy, the existing nodes should update the corresponding *CK*, *BK*, and *FK*. However, we use a lazy update strategy and do not update the keys immediately. We delay the key update until the BS wants to broadcast messages so that if there are other nodes joining the group, the rekey messages can be combined into one. At last the *CH* sends the corresponding keys encrypted with the joining node's *SK* to the new node(s). The *CH* sends new *CK* encrypted with corresponding *BK* to the nodes which are not in the same branch with the newly joining nodes. The *CH* sends new *CK* and *BK* encrypted with corresponding *FK* to the nodes which are in the same branch but have different parents with the newly joining nodes. The *CH* sends new *CK*, *BK* and *FK* encrypted with corresponding *SK* to the nodes which have the same parent with the newly joining nodes.

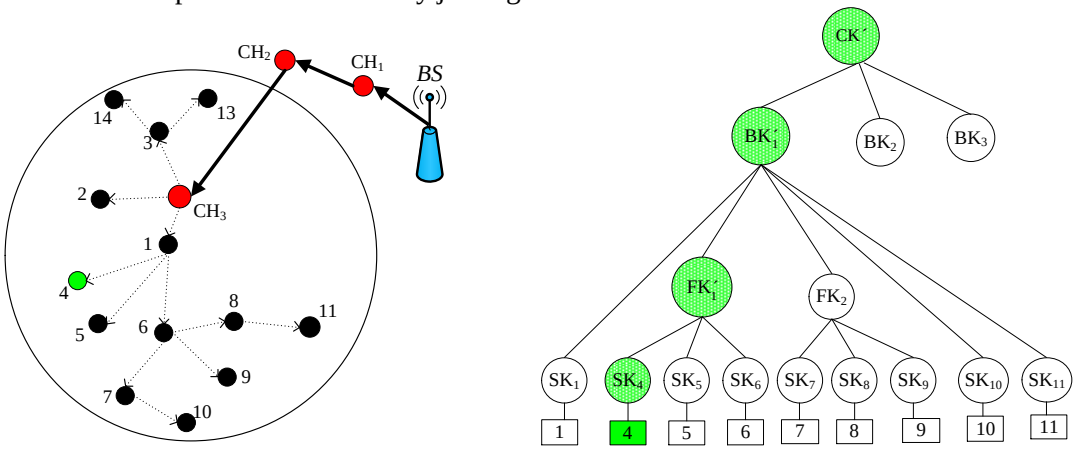

Fig. 6. Route tree and key tree after node 4 joins the cluster

Take *node* 4 in **Fig. 6** as an example. Suppose that *node* 4 wants to join the cluster. *Node* 4 broadcasts a *JoinRequest* message to the *CH*. After verifying the legitimacy of *node* 4, *CH* establishes a secret key SK_4 with it. And then, *CH* generates a new cluster key CK', a new branch key BK_1', a new fellow key FK_1'. The key update procedure is shown as follows: Nodes in branch 2 and branch 3 can update the new cluster key CK' by the message

$$CH \rightarrow Branch_i : \{update \,||\, E(CK', BK_i)\}, i = 2, 3.$$

Node 7, 8 and 9 can update the new cluster key CK', the new branch key BK_1' by a message

$$CH \rightarrow node\ 7, 8, 9 : \{update \,||\, E(\{CK', BK_1'\}, FK_2)\}.$$

Node 1, 10 and 11 can update the new cluster key CK', the new branch key BK_1' by the message

$$CH \rightarrow node\ i : \{update \,||\, E(\{CK', BK_1'\}, SK_i)\}, i = 1, 10, 11.$$

Node 4, 5 and 6 can update the new cluster key CK', the new branch key BK_1' and the new

fellow key FK_1' by the message

$$CH \rightarrow node\ i : \{update \,||\, E(\{CK', BK_1', FK_1'\}, SK_i)\}, i = 4, 5, 6.$$

After that, all the keys are updated securely. *Node* 4 becomes a member of the cluster and is able to decrypt the cluster messages.

3.4 Node Leaving

In wireless sensor networks, There are many reasons why a sensor node leaves the group. One of the reasons is that a node fails due to hardware failure or physical damage. Another reason is that the node is compromised by the adversary. Once the intruder is detected by means of some intrusion detection systems, the compromised node will automatically be removed from the group by key updating. In order to guarantee forward secrecy, the key tree should be updated as well. The probability of *CH* leaving event is much lower than the cluster member nodes.

1) *CH* Leaving: When a *CH* has not enough energy or is compromised, a new *CH* will be elected. The new *CH* will implement *CH* Joining procedure described in section 3.3 normally. And then, the *CH* Key Tree Generation procedure will proceed.

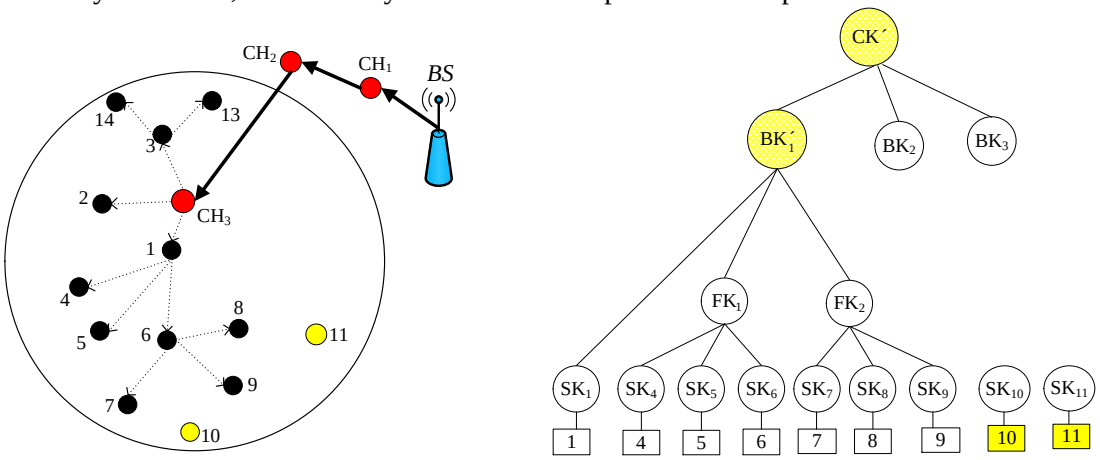

Fig. 7. Route tree and key tree after node 10 and 11 join the cluster

2) Cluster Member Node Leaving: There are two kinds of node leaving event: leaf node leaving event and non-leaf node leaving event. When the leaf node leaves the group, the topology of the rest of the nodes remains the same and the lazy update strategy is also used here. We delay the key update until BS wants to broadcast messages so that if there are other nodes leaving the group, the rekey messages can be combined into one. For example, in **Fig. 7**, *node* 10 leaves the cluster first, and *node* 11 leaves the cluster later. If there is no message transmitted in the cluster during this interval, two key update procedures can be merged into one. Before transmitting the data, *CH* will update the keys. *CH* generates a new cluster key CK', a new branch key BK_1'. The key update procedure is shown as follows:

Nodes in *Branch* 2 and *Branch* 3 can update the new cluster key CK' by the message

$$CH \rightarrow Branch_i : \{update \,||\, E(CK', BK_i)\}, i = 2, 3.$$

Node 4, 5 and 6 can update the new cluster key CK', the new branch key BK_1' by a message

$$CH \rightarrow node\ 4, 5, 6 : \{update \,||\, E(\{CK', BK_1'\}, FK_1)\}.$$

Node 7, 8 and 9 can update the new cluster key CK', the new branch key BK_1' by a message

$$CH \rightarrow node\ 7, 8, 9 : \{update \,||\, E(\{CK', BK_1'\}, FK_2)\}.$$

Node 1 can update the new cluster key CK', the new branch key BK_1' by a message

$$CH \rightarrow node\ 1 : \{update \,||\, E(\{CK', BK_1'\}, SK_1)\}.$$

When a non-leaf node leaving event occurs, the children of the leaving node become orphan nodes. Each orphan node will perform the same procedure as the Cluster Member Node Joining described in section 3.3 to join the cluster again.

4. Security analysis

In this section, we give a detailed security analysis of HRKT scheme. It shows that our scheme satisfies the requirements of group key distribution. Further analysis proves that HRKT scheme resists to Sybil attack, Denial of Service (DoS) *Rep* attack and sensor node compromise attack.

Theorem 1: HRKT scheme is a secure group key distribution.

Proof: HRKT scheme satisfies the requirements of group key distribution in the following aspects:

(1) Group confidentiality: In the HRKT scheme, it is difficult for an active attacker to compute the group key. The attackers can implement passive attacks such as eavesdropping, and/or active attacks such as inserting forged messages. But the group keys are only sent to legal nodes; hence, the attackers do not know any keys. The attackers cannot deduce the group key and plaintext from the ciphertext with non-negligible probability, since the functions $E(m, k)$ are secure against cipherext-only attacks. Therefore, the attackers are not able to obtain the group key in acceptable time.

(2) Backward secrecy: When a node joins the group, the proposed HRKT scheme updates all corresponding keys as section 3.3 describes. The newly joining node cannot obtain any information of the previous group keys. Hence, the probability of the newly joining node deriving previous keys is negligible.

(3) Forward secrecy: Assume that a node once was a group member and it had all the key encryption keys before it left the group. After it leaves the group, our HRKT scheme updates all corresponding keys as section 3.4 describes. The removed node cannot obtain any relevant keys any more. The probability of the removed node derives subsequent key encryption keys is negligible. Hence, the leaving nodes are not able to compute the new key in acceptable time.

In conclusion, HRKT scheme is a secure group key distribution. ■

Theorem 2: HRKT scheme can resist to Sybil attack, Denial of Service (DoS) *Rep* attacks and node compromise attack.

Proof:

(1) Protection against Sybil attack: Before a node joins the group, it must be verified by sending its *ID* and a *MAC* signature to the BS. As a result, the malicious nodes and compromised nodes trying to launch Sybil attacks by inventing many *IDs* will be excluded from the group.

(2) Protection against Denial of Service (DoS) *Rep* attack: Because the total number of *CHs* is small, an abnormally high flood of *Rep* messages can be easily detected by the BS. There are several ways to prevent such attack. In the HRKT scheme, *CHs* generate a sliding window to store the average number of *Rep* messages they have received within some time window. In case that this number exceeds a threshold, then *CH* will not forward any *Rep*

messages until the average number of sliding window falls well below the threshold. Therefore, DoS *Rep* attack can be mitigated in our scheme.

(3) Protection against node compromise attack: In the HRKT scheme, the *CH* will delete the seed and all the *BK* and *FK* after cluster member key generation. Even if the *CH* is compromised, the attacker cannot get all the keys. As a result, it mitigates the effect of compromised *CH*. The new keys will be generated once there is any change in the group membership (either join or leave). Once the attackers are detected by means of some intrusion detection systems (IDS), the compromised nodes will be automatically removed from the group by key update, which turns our scheme resilient to node capture attacks. ■

5. Performance evaluation

In this section, we give a detailed performance evaluation of the proposed HRKT scheme from the storage and communication point of view.

Storage cost: In the HRKT scheme, Each *CH* just needs to store at most two level keys, a *PK*, a *CK*, some *SKs* and several random numbers. The cluster member node just needs at most 5 keys (*CK*, *BK*, *FK*, *SK*, and *PK*). Hence, the storage requirement is lightweight in our scheme.

Communication cost: We have simulated HRKT in QualNet 5.0 on an Intel E4600 box running Windows XP. We compare our scheme to the traditional LKH scheme and SLIMCAST. The unit communication costs are set to $e_{tx} = 0.209\mu J$ and $e_{rx} = 0.226\mu J$, where e_{tx} is the energy dissipated during 1-*bit* transmission by a sensor node and e_{rx} is the energy dissipated during 1-*bit* reception by a sensor node [26]. The inter-cluster communication cost is 1.5 times larger than intra-cluster communication cost. The unit rekeying message size is set to 128 *bits* [19].

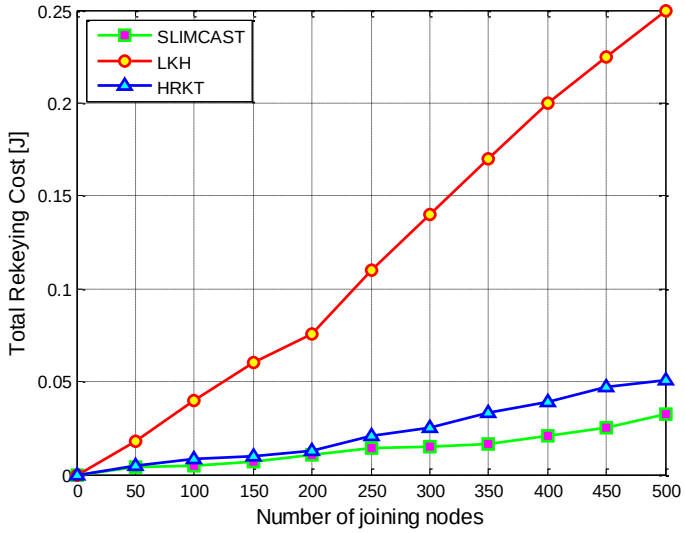

Fig. 8. Total rekeying cost according to the increasing number of joining nodes

Fig. 8 illustrates simulation results of total rekeying cost according to the increasing number of joining nodes. As the scale of the wireless sensor network becomes large, the number of level members in SLIMCAST and the number of *CH* in HRKT scheme will remain the same.

The number that will be affected is the average hop count between a newly joining node and the BS, and this only affects LKH scheme dramatically. This is the reason why the cost in LKH scheme grows much more rapidly than SLIMCAST scheme and HRKT scheme. The cost of HRKT scheme is a little more than SLIMCAST scheme. The reason is that HRKT scheme needs to generate and manage a route tree in each cluster, and consume more energy in hop-by-hop encryption. But with the route tree, we can achieve different ranges of group communication. Hop-by-hop encryption can improve the security level of communication among *CHs*. Therefore, the slight additional energy cost is worthwhile.

Fig. 9 illustrates simulation results of total rekeying cost according to the increasing number of leaving nodes. With the number of leaving nodes increasing, the topology of the network keeps changing. The group key has to be updated in the whole network whenever a node leaves the network. Thus, the cost of key update in LKH scheme and SLIMCAST scheme grows largely. In HRKT scheme, the network is divided into several clusters, and the cost of key update can be confined in a cluster. What is more, several key updates can be combined into one by using the lazy strategy. As a result, the cost of key update in HRKT scheme grows slowly.

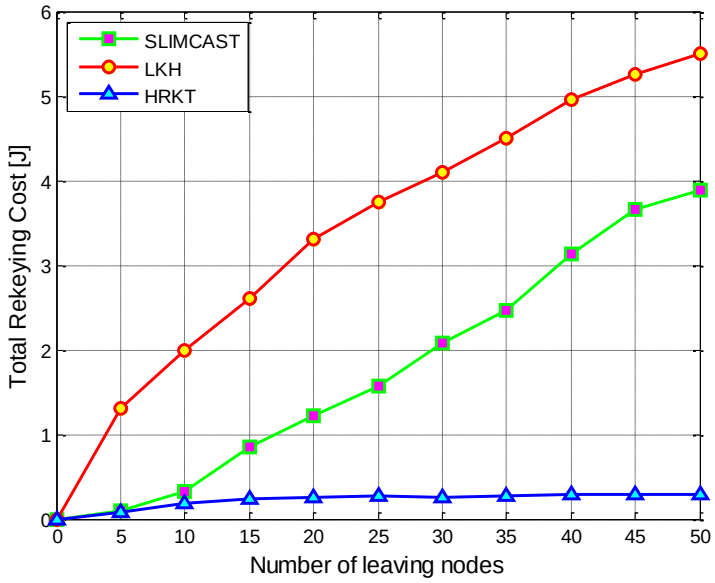

Fig. 9. Total rekeying cost according to the increasing number of leaving nodes

6. Conclusion

In this paper, we have proposed a hierarchical cluster-based secure and scalable group key management scheme for wireless sensor networks. Based on the fact that the security requirements of communication are different, two protocols for cluster heads and cluster member nodes are designed according to the route topology of the wireless sensor network. Since the nodes which are geographically close to each other in a level or in a cluster share the same key encryption keys, the number of key updating messages can be reduced. Furthermore, a lazy update strategy can be leveraged to reduce the rekeying cost. The requirement of key storage is low for each sensor node and meets the limited capability of the sensor nodes in HRKT scheme. The security analysis shows that the proposed scheme meets the security requirements of group communication from the setup phase to key update events and protects

the data and sensor nodes against several attacks. Our simulation results show that when membership changes are massive, HRKT scheme can achieve impressive energy savings that increase the network lifetime and improve the scheme scalability. In our future work, we will consider the case that the base station can be mobile, and then study dynamic participation mechanism, i.e., dynamically joining and revoking nodes.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications magazine*, vol. 40, no. 8, pp. 102-114, Aug. 2002. Article (CrossRef Link)

[2] X. Wang, J. Luo, Y. Liu, S. Li, and D. Dong, "Component-based localization in sparse wireless networks," *IEEE/ACM Transactions on Networking*, vol. 19, no. 2, pp. 540-548, Apr. 2011. Article (CrossRef Link)

[3] R. Deng, J. Chen, C. Yuen, P. Cheng, and Y. Sun, "Energy-Efficient Cooperative Spectrum Sensing by Optimal Scheduling in Sensor-Aided Cognitive Radio Networks," *IEEE Transactions on Vehicular Technology*, vol. 61, no.2, pp. 716-725, Feb. 2012. Article (CrossRef Link)

[4] P. Cheng, R. Deng, and J. Chen, "Energy-Efficient Cooperative Spectrum Sensing in Sensor-Aided Cognitive Radio Networks," *IEEE Wireless Communications*, vol.19,no.6, pp. 100-105, Dec. 2012. Article (CrossRef Link)

[5] R. Jiang, J. Luo, and X. Wang, "An attack tree based risk assessment for location privacy in wireless sensor networks," in *Proc. of 8th IEEE International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1-4, Sep. 21-23, 2012. Article (CrossRef Link)

[6] C. Lai, H. Li, Y. Zhang, and J. Cao, "Security issues on machine to machine communications," *KSII Transaction on Internet and Information Systems*, vol. 6, no. 2, pp. 498–514, Feb. 2012. Article (CrossRef Link)

[7] M. Wen, J. Li, Z. Yin, et al., "A NTRU Based Key Generation and Data Transmission Scheme for Sensor Networks," *Journal of Computational Information Systems*, vol. 8, no.6, pp. 2417-2424, June, 2012. Article (CrossRef Link)

[8] M. Wen, Y. Zheng, W. Ye, K. Chen, and W. Qiu, "A key management protocol with robust continuity for sensor networks," *Computer Standards & Interfaces*, vol. 31, no. 4, pp. 642–647, Apr. 2009. Article (CrossRef Link)

[9] C. Lai, H. Li, X. Li, and J. Cao, "A novel group access authentication and key agreement protocol for machine-type communication," *Transactions on Emerging Telecommunications Technologies*, 2013. Article (CrossRef Link)

[10] R. Lu, X. Lin, H. Zhu, X. Liang, and X. Shen, "BECAN: A Bandwidth-Efficient Cooperative Authentication Scheme for Filtering Injected False Data in Wireless Sensor Networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 23, no. 1, pp. 32-43, Jan. 2012. Article (CrossRef Link)

[11] S. Rafaeli and D. Hutchison, "A survey of key management for secure group communication," *ACM Computing Surveys*, vol. 35, no. 3, pp. 309-329, Sep. 2003. Article (CrossRef Link)

[12] R. Jiang, J. Luo, F. Tu, and J. Zhong, "LEP: A lightweight key management scheme based on EBS and polynomial for wireless sensor networks," in *Proc. of IEEE International Conference on Signal Processing, Communications and Computing*, pp. 1-5, Sep. 14-16, 2011. Article (CrossRef Link)

[13] C. K. Wong, M. Gouda, and S. S. Lam, "Secure group communications using key graphs," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16-30, Feb. 2000. Article (CrossRef Link)

[14] L. Lazos and R. Poovendran, "Secure broadcast in energy-aware wireless sensor networks," *DTIC Document*, Tech. Rep., 2002. Article (CrossRef Link)
[15] R. Di Pietro, L. V. Mancini, Y. W. Law, S. Etalle, and P. Havinga, "LKHW: A directed diffusion-based secure multicast scheme for wireless sensor networks," in *Proc. of IEEE International Conference on Parallel Processing Workshops*, pp. 397-406, Oct. 6-9, 2003. Article (CrossRef Link)
[16] J.-H. Huang, J. Buckingham, and R. Han, "A level key infrastructure for secure and efficient group communication in wireless sensor network," in *Proc. of the IEEE First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, pp. 249-260, Sept. 2005. Article (CrossRef Link)
[17] J.-H. Son, J.-S. Lee, and S.-W. Seo, "Topological key hierarchy for energy-efficient group key management in wireless sensor networks," *Wireless personal communications*, vol. 52, no. 2, pp. 359-382, Jan. 2010. Article (CrossRef Link)
[18] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks," *Wireless networks*, vol. 8, no. 5, pp. 521-534, Sept. 2002. Article (CrossRef Link)
[19] Y. Wang, G. Attebury, and B. Ramamurthy, "A survey of security issues in wireless sensor networks," *IEEE Communications Surveys Tutorials*, vol. 8, no. 2, pp. 2-23, Second Quarter 2006. Article (CrossRef Link)
[20] M. Shi, X. Shen, Y. Jiang, and C. Lin, "Self-healing group-wise key distribution schemes with time-limited node revocation for wireless sensor networks," *IEEE Wireless Communications*, vol. 14, no. 5, pp. 38-46, Oct. 2007. Article (CrossRef Link)
[21] J. Polastre, R. Szewczyk, C. Sharp, and D. Culler, "The mote revolution: Low power wireless sensor network devices," in *Proc. of IEEE HotChips 16*, vol. 16, pp. 22-24, 2004. Article (CrossRef Link)
[22] D. Wei, S. Kaplan, and H. A. Chan, "Energy efficient clustering algorithms for wireless sensor networks," in *Proc. of IEEE International Conference on Communications Workshops*, pp. 236-240, May 19-23, 2008. Article (CrossRef Link)
[23] D.-Y. Kim, J. Cho, and B.-S. Jeong, "Practical data transmission in cluster-based sensor networks." *KSII Transact0ions on Internet and Information Systems (TIIS)*, vol. 4, no. 3 pp. 224-242, June 2010. Article (CrossRef Link)
[24] R. Lu, X. Lin, H. Zhu, and X. Shen, "TESP2: Timed efficient source privacy preservation scheme for wireless sensor networks," in *Proc. of IEEE International Conference on Communications (ICC)*, pp. 1–6, May 23-27, 2010. Article (CrossRef Link)
[25] X. Lin, R. Lu, and X. Shen, "MDPA: Multidimensional Privacy-Preserving Aggregation Scheme for Wireless Sensor Networks," Wireless Communications and Mobile Computing (Wiley), vol. 10, no. 6, pp. 843-856, June, 2010. Article (CrossRef Link)
[26] T. I. Inc, "Single-chip 2.4GHz IEEE 802.15.4 compliant and Zigbee (TM) ready RF transceiver," Tech. Rep., 2007. Article (CrossRef Link)

Rong Jiang received the B.S. and M.S. degrees in 2007 and 2009, respectively, from the School of Computer Science, National University of Defense Technology, Changsha, China, where he is currently pursuing the Ph.D. degree. He is now a joint Ph.D student at Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research interests include wireless sensor networks security and privacy preservation, cloud computing and smart grid.

Jun Luo received the B.S. degree from the Computer School, Wuhan University, Wuhan, China, in 1984, and the M.S. degree from the School of Computer Science, National University of Defense Technology, Changsha, China, in 1989. He is now a Professor in the School of Computer Science, National University of Defense Technology. His research interests include operating system, cloud computing, information security, and sensor networking.

Xiaoping Wang received his BS, MS, and Ph.D degree in the School of Computer Science from National University of Defense Technology, China, in 2003, 2006, and 2010, respectively. He is now an assistant professor in the School of Computer Science at National University of Defense Technology. His research interests include parallel and distributed computing.