Multihop-Based Key Management in Hierarchical Wireless Sensor Network

Yiying Zhang[1,3,*], Xiangzhen Li[2], Yan Zhen[2], and Lingkang Zeng[2]

[1] State Grid Information & Telecommunication Company Ltd., Beijing, 100761, China
[2] State Grid Electric Power Research Institute, Nanjing 210003, China
[3] Beijing University of Posts and Telecommunications, Beijing, 100876, China
winzyy@163.com, {xzli,zhenyan,lkzeng}@sgcc.com.cn

Abstract. The vulnerable environment and open communication channel make it very necessary to protect wireless sensor networks (WSNs). The key management is the most important way to protect the communication in WSNs. In this paper, we design a Multihop-based Key Management (MKM) which can efficiently enhance the security and survivability in the hierarchical WSNs. Different from previous works, we present the key system as well as the cluster formation. The MKM generates and distributes keys based on hop counts, which not only localizes the key things but also has no overhead. The MKM provides the session keys among sensors and the cluster key between the cluster head and its member nodes. The different hops make the different keys. The MKM can protect the network from the compromised nodes by reducing the high probability of the common keys. The security analysis can effectively prevent several attacks.

Keywords: Key management, wireless sensor network, security, multi-hop.

1 Introduction

The wireless sensor networks (WSNs) are wildly used with thousands of tiny sensors, such as in smart grid, smart city, internet of things (IoTs). However, due to limitations of wireless sensor networks in the computation, energy, storage and open wireless communication etc., WSNs are vulnerable to various attacks, and the security in WSNs is required [1, 2, 3, 4, 5, 6]. Since it is impossible and impracticable to utilize the single key to encrypt/decrypt message in the networks, the solutions, which trend to deploy keys in whole network, may cause the leak of key materials. Once the key things fall into the adversaries, the sensors are compromised, which threatens entire network.

Therefore, it is necessary to organize the sensors into clusters and localize the key things. In [2, 3], the authors presented RPKH and LDK schemes to provide the local key management. The RKPH and LDK utilize different nodes including the normal

* Corresponding author.

R. Li, J. Cao, and J. Bourgeois (Eds.): GPC 2012, LNCS 7296, pp. 302–311, 2012.

nodes and anchor nodes to generate keys by different transmission range. However, it also consumes large amount of energy, when the two kinds of nodes transfer messages and discovery common keys. Meanwhile, the adversary can eavesdrop on the key materials during nodes exchanging packets.

In this paper, we present a Multihop-based Key Management (MKM) in the hierarchical wireless sensor network [8]. Different from the previous works, we build our solution in normal network without any special nodes (e.g. high energy or high capability nodes), which makes it more practicable to deploy. In MKM, we generate the key system with the hierarchical architecture formation, which makes the scheme effective because of no overhead. When the clusters form, the cluster head gets the hop counts from cluster head to the member nodes with ACK packets and then uses the hop-count and nonces to generate the key system. Nodes in different distance have different keys.

In MKM, the nodes near the cluster head have more keys, which makes the nodes transfer messages on one-way routing. Meanwhile, with the cluster head reselection, the key system will be refreshed, because the new cluster head causes the hops changes and then the key should be reassigned. Moreover, the cluster head can use the cluster key to communicate with member nodes.

Considering about the security and the life time of WSNs, we will rekey to refresh the cluster and the keys. During the rekey phase, the cluster will elect new cluster head which calculate the new distance from CH to member nodes and then generate the new key system based on the old one.

Compared to previous works [2, 3, 5] in WSNs, our solution has the following scientific research contributions: 1) MKM utilizes the hierarchical architecture to localize the key things, which prevents the compromised nodes threat the entire network. 2) Without any overhead, MKM counts the hop count in the cluster formation, which can effectively reduce energy consumption. 3) MKM employs the normal wireless sensor network but not special nodes, which makes it more practicable.

The rest of this paper is organized as follow: Section 2 presents related work. Section 3 shows the system model. Section 4 will describe the key management in detail, and section 5 evaluates MKM using security analysis. Finally, we end the paper with a conclusion as well as the further work.

2 Related Work

Some literatures are researched in [2, 3, 5, 13, 14], these papers designed some schemes for local key management.

In [3], a location dependent key management (LDK) has been presented, which employs the heterogeneous sensors to build a clustered sensor network. In LDK, the sensors are static and considered the anchor nodes as the management nodes. The anchor nodes use the different location information to generate sets of keys. The adjacent nodes subjected to the same anchor node can establish secure communication links by exchanging all key materials to discover the common keys. Neighbouring

nodes can establish secure communication link by determining the common keys via exchanging their key materials. LDK significantly reduces the number of keys stored on each node without any pre-deployment knowledge of nodes. It also can increase the direct connectivity ratio among nodes.

However, the LDK makes nodes exchange all their key materials when determining the communicating key, which consumes more energy and increase attack chances. The transmitted message takes too many bytes. Therefore, it consumes lots of communicating energy and is not efficient for WSNs either.

In [2, 5], there are two similar key management schemes, the RPKH and ARPKH. The RPKH is based on random key distribution in the heterogeneous sensor networks, which used separate keys in different clusters and take into consideration distance of sensors from theirs cluster head.

The ARPKH, an improved version of RPKH, considers a multiple shared keys between pair-wise nodes on connectivity. When a key that used for establishing the secure link between two nodes is revealed, the link has been expired and then the connectivity is broken. ARPKH will change the alternative shared keys to replace the revealed key and establishes a new secure link between two nodes again.

Chan et al. [14] proposed q-composite random key pre-distribution scheme which needs q common keys ($q > 1$) to establish secure communication links between nodes. By increasing the value of parameter q, they can achieve high security level, because an adversary needs to know all q keys for attack. The drawback of this scheme is that if an adversary has compromised nodes in large scale, entire keys can be exposed.

In wireless sensor networks, nodes need to communicate with neighboring nodes only. Therefore, it can be more efficient to let nodes which are close together have common keys while between nodes which are far off have disjointed keys. Thus, W. Du et al [13] proposed a scheme that uses pre-deployment knowledge of expected location of nodes. Through allocating keys among adjacent nodes by using the pre-deployment knowledge, the memory requirement on a node can be decreased and the security impact of compromised nodes can be limited around the location.

3 System Model

3.1 Network Model

Given the WSN with n sensors as a graph G which consists of m clusters, that is, $G = C_1 \cup C_2 \ldots \cup C_m$ and $C_i \cap C_j = \phi, i \neq j$, where C_i is a cluster with the cluster head(CH or CH_i) and member nodes. In a cluster, the *CH* collects and aggregates packets from its member nodes, and then forwards them to the base station (BS). Normally, a member sensor can transfer packets to CH through several hops. Note that there are some nodes in the range overlap of many cluster, they only choose one cluster to join. Each sensor has a unique ID. If a node is compromised, all of the information in this node will be compromised including the key materials [7].

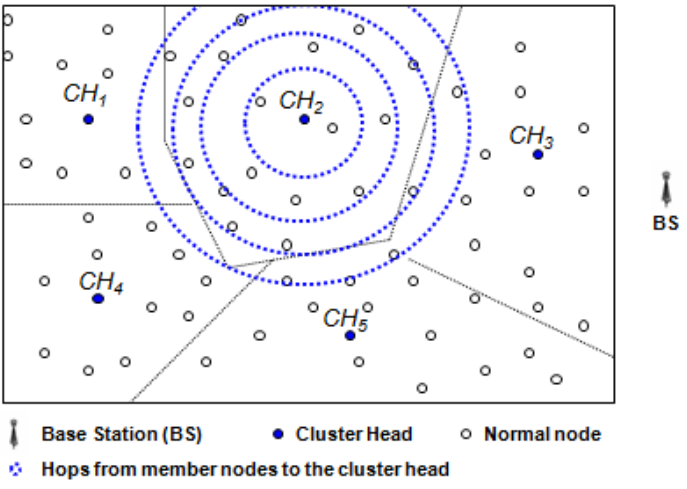

Fig. 1. The considered hierarchical WSN with 5 clusters

3.2 Notations

In Table 1, we list some notations used in this paper.

Table 1. Notations

Notation	Explains
K_I	The initial key shared by all nodes
ID_{CH}	ID of the cluster head
n_i	The i^{th} nonce in the set of nonces N
ID_{v_j}	ID of member node v_j
$f()$	The *one-way* function
TTL	Time To Live
C_i	The i^{th} cluster in the WSN
k_j^i	The i^{th} key for the member node v_j

4 Multihop-Based Key Management

In this section, we introduce the Multihop-based key management (MKM) in detail. Before the deployment of the sensor network, each sensor is pre-distributed an initial key K_I for the security in deployment phase, the initial key provides the communication during the formation phase [7].

4.1 The Cluster Head Election

As mentioned above, considering the energy efficiency and management facilitation of WSN, we adopt the hierarchical architecture for our network [9, 10, 11]. Firstly, a node itself decides whether it becomes a candidate CH or not according to the cluster head election algorithm as shown in equation (1):

$$T(n) = \begin{cases} \dfrac{p}{1-p[r \mod (1/p)]}[\eta+(i\times p)(1-\eta)], & n\in G \\ 0, & otherwise \end{cases} \quad (1)$$

Where $T(n)$ indicates a probability function; p is the percentage; and r denotes the current round number and its default value is 1; η represents the left energy percentage; i is the sum of rounds that a node is idle, and it is reset to 0 when it is elected as CH. G is the set of nodes that have not been CHs in the 1/p rounds recently.

The node will announce the candidate information to other nodes according to the result of equation (1). And other nodes which maybe accept several election campaign messages, and they will choose one to join it as follow.

4.2 The Cluster Formation

Once a node becomes a cluster head, it will send a *beacon* message to other sensors to form a cluster. Each sensor may receive several different *beacon* messages from different candidate cluster heads, but it only can join one cluster.

When the *CH* broadcasts a *beacon* message encrypted by K_1 contains ID of *CH*, *TTL* (time to live) and a set of *nonces* named *N*, the random numbers. And the *nonce*s will be different with different *TTL*, that is, if the *TTL*= 3, then the sensor will get four (*TTL*+1) nonces, such as N={ n_1, n_2, n_3, n_4}. We generate more nonces (e.g. *TTL*+1) for the connectivity, especially the common keys. Where *TTL* is to limit the cluster size, e.g. *TTL*=3, and it will be decrease for each forwarding until it becomes 0 and the beacon message will be dropped.

Therefore, depending on the cluster size (*TTL*), other nodes can receive different sets of *beacon* messages from different *CHs* as the equation (2) in different distance (hop ranges) .

$$CH \rightarrow *: \{ID_{CH}, n_1 \ldots n_{TTL+1}, TTL\}_{K_1} \quad (2)$$

4.3 The Initial Key Generation

Assume $v_j \in C_i, v_j \neq CH$, we call v_j as member node. When member node v_j receives a *beacon* message and wants to join the cluster C_i, it counts the TTL and sends the ACK including the ID_{v_j} and the TTL_{v_j} back to its interest cluster, and then the cluster head knows the hops from ID_{v_j} to CH. *Beacon* messages are orderly transmitted at

different distance levels. And then, the member node v_j decrypts the *beacon* messages and obtains ID_{CH} and then set of nounces N_i.

And then, v_j calculates the candidate keys k_j^i based on the received *nonce*s as follows:

$$k_j^i = f(k_I, n_i) \tag{3}$$

Where *f*() denotes a one-way hash function. And the initial key generation process is as shown in fig.2.

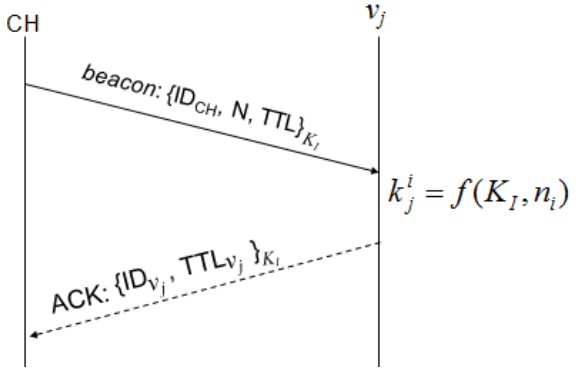

Fig. 2. The initial key generation process

After the calculations, nodes erase K_I. Consequently, v_j stores the key things as follows in table 2:

Table 2. Keys table of member nodes in different hop ranges (*TTL*=3)

Keys for 1st hop	Keys for 2nd hop	Keys for 3rd hop
$k_j^1, k_j^2, k_j^3, k_j^4$	k_j^2, k_j^3, k_j^4	k_j^3, k_j^4
$k_j^1, k_j^2, k_j^3, k_j^4$	k_j^2, k_j^3, k_j^4	k_j^3, k_j^4
…	…	…
$k_j^1, k_j^2, k_j^3, k_j^4$	k_j^2, k_j^3, k_j^4	k_j^3, k_j^4

According to the table 2, we can find that the nodes can communicate with its neighbour nodes for the common keys. The specific algorithm of hop count and key information acquirement is as algorithm 1:

Table 2. Hop count and key information acquirement algrithm (*TTL*=3)

Algorithm 1 Hop count and key information acquirement	
1.	*CH* broadcasts *beacon* messages with different *nounces*: $CH \rightarrow *: \{ID_{CH}, n_1 ... n_{TTL+1}, TTL\}_{K_I}$
2.	$v_j \rightarrow CH: \{ID_{v_j}, TTL_{v_j}\}_{K_I}$, *Key pool generation for* v_j by $k_j^i = f(k_I, N_i)$
3.	*Erease* K_I.
4.	CH obtains the hop count: $N_{hops} = TTL_{CH} - TTL_{v_j} + 1$
5.	According to the N_{hops}, nonces N, and oneway function f(), the cluster head can get the ID_{v_j}'s key information.
6.	**end**.

4.4 The Common Key Discovery

For communication, member node should establish secure link with its neighbouring nodes, which needs the common keys between them. According to those candidate keys, member nodes in the same distance receive the same *beacon* messages and they can also generate the same keys. Moreover, the nodes in the adjacent areas also have some duplicate candidate keys.

If a node can receive $\{ID_{CH}, N_i, TTL_i\}_{K_i}$, it also can receive $\{ID_{CH}, N_j, TTL_j\}_{K_i}$, where $TTL_i > TTL_j$ ($i<j$). Since the distance range of hops j covers the distance range of hops i, the node near cluster head has more keys than the one far away from cluster head.

Therefore, each member node v_j generates a list which just stores the keys .Moreover, since the packets from members will be collected by the cluster head, the cluster head should have the ability to decrypt these messages. During this process, the member nodes should report its keys, which will increase the transmission. Because the nonces are sent by the cluster head, it also knows the function, and then it can calculate the keys of members as mentioned in algorithm 1.

Due to the keys are generated by hop count, which means the nodes can be connected in the same cluster. And the path key between v_i and v_j is calculated as follow:

$$k_{ij} = f^{abs(i-j)}(k_I, N_i) \tag{4}$$

The equation (4) makes it possible for any two nodes in the cluster to communicate with each other. Actually, there is another way to make every two nodes communicate, that is, the last but one nonce is same in a cluster and the key which they generate is same too (the last nonce is used to the cluster key).

The cluster key is the key which is used for communication between the CH and its members also for generating new key in next round. Since there are TTL+1

nonces, according to the algorithm 1, the last nonce in the set N will be transferred to every sensor.

4.5 The Rekey Process

For prolonging the lifetime of the whole network, it is necessary to change the cluster head. On the other hand, the key should be rekeyed for the security [12], otherwise, when *CH* receives a certain amount of encrypted messages (more than $2^{2k/3}$, k is the length of key), the keys will be no longer safe. According to the requirements abover, we should recluster after a certain phase. Assume the process of recluster happens inside the cluster, which can reduce the energy consumption.

During the reselection of CH, we can rekey as the initial phase. When the new cluster head has been selected according to the equation (1), it will announce itself as the cluster head and recalculate the distance from its members.

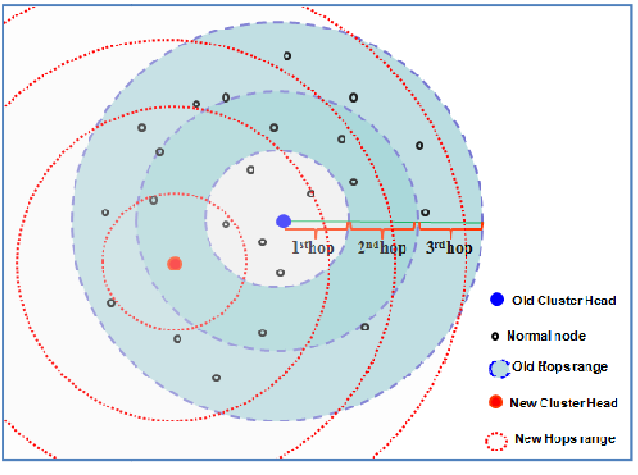

Fig. 5. The reselection of CH

As shown in fig 5, in this situation, the new cluster head changes not only the relative position but also hop counts from CH to members, which make the nounces as well as the key things different.

5 Security Analysis

Compared to previous works, the salient advantage of our solution is that we addressed challenging runtime security issues using localizing key things and design a dynamic key management.

During cluster formation phase, the cluster head can calculate the hop count from the CH and member nodes, and then the member nodes can generate keys by the

nonces and hops. According to the different hop count, the cluster is divided into several security belts as shown in fig 2, the nodes in different belts have different keys. Because the keys are generated by the set of nonces, the adjacent nodes have some common keys, which makes it possible to communicate with each other.

The nodes near CH have more keys than the nodes far away from CH, which means that the far nodes just can submit message to the CH. And then the messages just can be decrypted by near nodes. The one-way security model prevents the eavesdrop attack, selective-forwarding attack and hello flood attack as shown in Table 3.

Table 3. Analysis in local key management

Attacks Types	RPKH	LDK	MKM
Selective-Forwarding	×	×	√
Sink-Hole attack	×	√	√
Sybil attack	√	√	√
Worm-Hole	√	√	√
HELLO Flood	√	√	√
DoS	×	×	√

To communicate with members, the cluster head utilize the last key as the cluster key which is shared with all the sensors (including the CH) in the cluster. The cluster also can be used to rekey during the next round cluster. Furthermore, the key system forms during the cluster formation, which almost does not consume any energy overhead.

6 Conclusion and Future Work

In this paper, we propose a Multihop-based key management (MKM) protocol to enhance network security and survivability. Unlike previous works, we employ the hierarchical architecture but not fixed-node-network. In contrast to other clustered architectural security solutions, the salient advantage of this work is that we addressed challenging security issues by localizing key things. Also we present a rekey mechanism with low energy consumption. In the future, we will focus on how to enhance security in scalable WSN.

Acknowledgements. This work was supported by the foundation: Important National Science & Technology Specific Projects of China: Research, development, and application validation of sensor network for smart grid security monitoring, transmission efficiency, measurement and user interaction (2010ZX03006-005-02); the National Basic Research Program of China (973 Program): Basic theory and practice research of Internet of Things (2011CB302900); the Doctoral Start-up Fund of Liaoling Province (20101074).

References

[1] Zhou, Y.: Securing Wireless sensor networks: A Survey. IEEE Communications Surveys & Tutorials (3rd Quarter, 2008)

[2] Banihashemian, S., Ghaemi Bafghi, A.: A new key management scheme in heterogeneous wireless sensor networks. In: Proceeding ICACT 2010, Korea (2010)

[3] Anjum, F.: Location dependent key management in sensor networks using without using deployment knowledge. In: Proceedings of WiSe 2006 (2006)

[4] Du, X., Xiao, Y., Guizani, M., Chen, H.-H.: An effective key management scheme for heterogeneous sensor networks. Ad Hoc Networks 5(1), 24–34 (2007)

[5] Banihashemian, S., Bafghi, A.G.: Alternative shared key replacement in heterogeneous wireless sensor networks. In: Proceeding IEEE Computer Society (2010)

[6] Luk, M., Mezzour, G., Perrig, A., Gligor, V.: MiniSec: A Secure Sensor Network Communication Architecture. In: IPSN 2007 (April 2007)

[7] Zhu, S., Setia, S., Jaodia, S.: LEAP+: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. ACM Transactions on Sensor Networks 2(4), 500–528 (2006)

[8] Abbasi, A.A., Younis, M.: A survey on clustering algorithms for wireless sensor networks. Computer Communications 30, 2826–2841 (2007)

[9] Bandyopadhyay, S., Coyle, E.J.: An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks. In: Proceeding of IEEE INFOCOM 2003, San Francisco (April 2003)

[10] Handy, M.J., Haase, M., Timmermann, D.: Low energy adaptive clustering hierarchy with deterministic cluster-head selection. In: Mobile and Wireless Communications Networks, pp. 368–372. IEEE Communications Society, Stockholm (2002)

[11] Manjeshwar, A., Grawal, D.P.: TEEN: A protocol for enhanced effciency in wireless sensor networks. In: PDPS 2001, pp. 2009–2015. IEEE Computer Society, San Francisco (2001)

[12] Abdalla, M., Bellare, M.: Increasing the Lifetime of a Key: A Comparative Analysis of the Security of Re-keying Techniques. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 546–559. Springer, Heidelberg (2000)

[13] Du, W., Deng, J., Han, Y., Chen, S., Varshney, P.: A key management scheme for wireless sensor networks using deployment knowledge. In: INFOCOM 2004: Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, vol. 1. IEEE (2004)

[14] Chan, H., Perrig, A., Song, D.: Random key predistribution schemes for sensor networks. In: Proceedings of Symposium on Security and Privacy, pp. 197–213. IEEE (2003)