Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2012, Article ID 547471, 8 pages
doi:10.1155/2012/547471

Research Article

A Secure Hierarchical Key Management Scheme in Wireless Sensor Network

Yiying Zhang,[1,2] Xiangzhen Li,[3] Jianming Liu,[2] Jucheng Yang,[4] and Baojiang Cui[1]

[1] *School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2] *Department of Production & Technology, State Grid Information & Telecommunication Company Ltd., Beijing 100761, China*
[3] *IoT Center, State Grid Electric Power Research Institute, Nanjing 210003, China*
[4] *College of Computer Science & Information Engineering, Tianjin University of Science & Technology, Tianjin 300222, China*

Correspondence should be addressed to Yiying Zhang, zhangyiying1973@hotmail.com

Received 12 June 2012; Revised 28 July 2012; Accepted 12 August 2012

Academic Editor: Hailun Tan

Due to the vulnerable environment, limited recourse and open communication channel, wireless sensor networks (WSNs) are necessary to be protected from various attacks. The key management is an important way to protect the communication in WSNs. In this paper, we present a hierarchical key management scheme (HKMS) which can efficiently enhance the security and survivability for the clustered WSNs. Different from previous works, the HKMS distributes keys based on hop counts and one-way function by the clustered architecture, which not only localizes the key things but also has no overhead. The HKMS provides the session keys among sensors and the cluster key between the cluster head and member nodes. The HKMS dynamically generates different keys based on different hops in different periods which can protect the network from the compromised nodes and reduce the high probability of the common keys without any special sensors (such as the anchor nodes). The security analysis and simulation show the HKMS can prevent several attacks effectively and reduce the energy consumption.

1. Introduction

Wireless sensor network (WSN) is usually considered as a large-scale network with thousands of tiny sensors and deployed in smart grid, smart city, smart home, and so forth to sense the information. As the most important part in the perception layer of internet of things (IoTs), WSNs are deployed for sensing, monitoring, or controlling various objects [1, 2]. However, there are still some limitations, such as the capability of computation, low energy, small storage and open communication channel. Therefore, WSNs are vulnerable to various attacks, and the security in WSNs is required [3–8].

Some literatures focus on localizing the key things. In [2, 3], the authors presented RPKH and location-dependent key management (LDK) schemes to provide the local key management. The RKPH and LDK utilize different nodes including the normal nodes and anchor nodes to generate keys by different transmission ranges.

In [3], LDK has been presented and it employs the heterogeneous sensors to build a clustered sensor network. In LDK, there are higher ability nodes, the anchor nodes, as the management nodes. The anchor nodes use the different location information to generate sets of keys. Neighboring nodes can establish secure communication link by determining the common keys via exchanging their key materials. LDK takes advantage of relative location of nodes after deployment by utilizing anchor nodes at different power levels. Based on different locations, nodes can receive different sets of keys from anchor nodes. Neighboring nodes can establish secure communication link through the common keys. LDK can increase the direct connectivity ratio among nodes. However, in LDK, nodes need to transmit a message that consists of all the key materials when determining common keys to establish secure links. Therefore, it consumes lots of communicating energy and is not efficient for WSNs, and the adversary also can eavesdrop on the key materials during nodes exchanging packets. Moreover, the special node (anchor node) makes it difficult to deploy.

In [5], the ARPKH is designed based on random key distribution in the heterogeneous sensor networks, which

uses separate keys in different clusters and take into consideration distance of sensors from theirs cluster head. Compared with the RPKH, the ARPKH considers a multiple shared keys between pairwise nodes. 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. Moreover, ARPKH will change the alternative shared keys to replace the revealed key and establish a new secure link between two nodes again. However, the ARPKH needs alternative shared key replacement, which makes sensors predistribute more keys and occupy larger storage. Moreover, ARPKH also needs the anchor nodes as the cluster head, which makes it impracticable.

In this paper, we present a hierarchical key management scheme (HKMS) in the clustered wireless sensor network [9]. Different from the previous works, our network needs no any special nodes (e.g., high-energy or high-capability nodes), which makes it more practicable to deploy. Meanwhile, HKMS also distributes keys through the key seed (nonce) according to TTL (time to live), which has higher level security than previous works transferring key things directly.

The HKMS builds the key system with the clustered architecture formation. The cluster head gets the hop counts from cluster head to the member nodes with ACK packets and then uses the hop count to determine TTL as well as a certain numbers of nonces for building the key system. During the key distribution, nodes in different hop ranges will be obtained different keys. With the cluster head reselection, the key system will be rebuilt, and then the key should be reassigned.

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 calculates the new distance from CH to member nodes and then generate the new key system based on the old one.

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

The rest of this paper is organized as follows Section 2 presents the system model. Section 3 describes the key management in detail. Section 4 evaluates HKMS using security analysis, meanwhile, we simulate the solutions to evaluate the performances of HKMS. Finally, we end the paper with a conclusion as well as the further work in Section 5.

2. System Model

2.1. Network Model. Given G is a WSN which consists of m clusters, that is, $G = C_1 \cup C_2 \cdots \cup C_m$ 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 [10, 11]. 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. Assume a normal node $v_j^i \in C_i$, it communicates with CH through multihop, as shown in Figure 1.

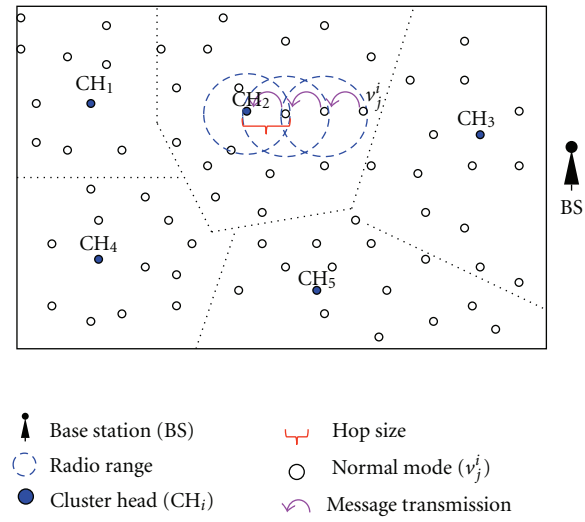

Figure 1: The considered wireless sensor network.

Table 1: Notations.

Notation	Explanation
K_I	The initial key shared by all nodes
ID_{CH}	ID of the cluster head
n_i	The ith 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 ith cluster in the WSN
k_j^i	The ith key for the member node v_j
v_j^i	The member node in C_i

2.2. Assumptions. In our network, all sensor nodes are deployed in the network uniformly and randomly and are static. Each sensor has a unique ID. If a node is compromised, all of the information in this node will be revealed including the key materials [12]. The sensors in network should be in at least one cluster.

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

3. The Hierarchical Key Management Scheme

In this section, we introduce the hierarchical key management scheme (HKMS) in detail. Before the deployment of the sensor network, each sensor is predistributed an initial key K_I for the security in deployment phase, the initial key provides the communication during the formation phase and will be erased after key deployment [12].

3.1. The Cluster Head Election. As mentioned above, considering the energy efficiency and management facilitation of

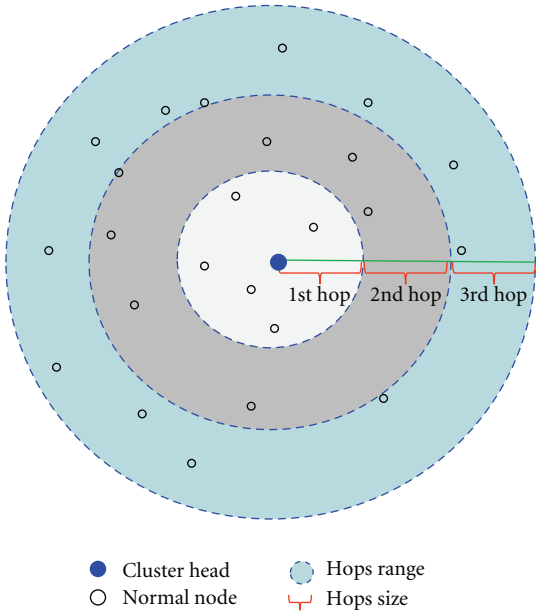

Figure 2: The nodes in different hop range to CH (assume these nodes join in the same cluster, TTL = 3).

WSN, we adopt the hierarchical architecture for our network [10, 11, 13]. Firstly, a node itself decides whether it becomes a candidate CH or not according to the cluster head election algorithm [10, 14]. The node will announce the candidate information to other nodes. And other nodes which may accept several election campaign messages, and they will choose one to join it as follows.

3.2. 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_I contains ID of CH, the transmission range is limited by TTL, that is, calculated by hop count. Usually, TTL is the hop count plus one. And then, TTL and a set of *nonces* named N, the random numbers. And the *nonces* 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, for example, TTL = 3, and it will be decreased 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 (1) in different distance (hop ranges) as shown in Figure 2:

$$N = \{n_i \mid n_1, \ldots, n_{\mathrm{TTL}+1}\}$$
$$\mathrm{CH} \longrightarrow * : \{\mathrm{ID}_{\mathrm{CH}}, N, \mathrm{TTL}\}_{K_I}. \quad (1)$$

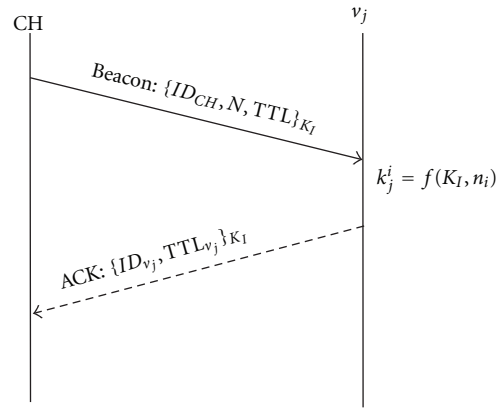

Figure 3: The initial key generation process.

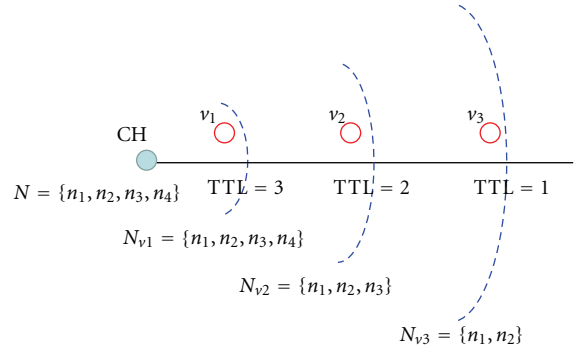

Figure 4: The deployment of nonces from the cluster head (TTL = 3).

3.3. The Initial Key Generation. Assume $v_j \in C_i$, $v_j \neq \mathrm{CH}$, 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.

The *beacon* messages are orderly transmitted at different distance levels. And then, the member node v_j decrypts the *beacon* messages and obtains $\mathrm{ID}_{\mathrm{CH}}$ and then set of *nonces* N_i.

And then, v_j calculates the candidate keys k_j^i and gets the key set K^i based on the received *nonces* as follows:

$$K^i = \left\{k_j^i \mid k_1^i, \ldots, k_{\mathrm{TTL}_{v_j}}^i\right\}, \quad k_j^i = f(k_I, n_i), \quad (2)$$

where $f()$ denotes a one-way hash function. The specific algorithm of key distribution is designed as in Algorithm 1.

And the initial key generation process is as shown in Figures 3 and 4.

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

According to 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 in Algorithm 2.

(1) CH broadcasts *beacon* messages with different *nonces*:
CH $\rightarrow * : \{\mathrm{ID}_{\mathrm{CH}}, N, \mathrm{TTL}\}_{K_I}$
(2) v_j decrypts$\{\mathrm{ID}_{\mathrm{CH}}, N, \mathrm{TTL}\}_{K_I}$
(3) $K^i = \phi$
(4) $k_{\mathrm{length}} = \mathrm{TTL}_{v_j^i}$
(5) *For* $j = 1$ *to* k_{length} {
$k_j^i = f(k_I, n_i)$ // generate the key pool
$K^i = K^i \cup \{k_j^i\}$
}
(6) *Erase* K_I.
(7) *end*.

Algorithm 1: Key distribution algorithm (TTL = 3).

(1) CH broadcasts *beacon* messages with different *nonces*:
CH $\rightarrow * : \{\mathrm{ID}_{\mathrm{CH}}, n_1 \cdots n_{\mathrm{TTL}+1}, \mathrm{TTL}\}_{K_I}$
(2) $v_j \rightarrow \mathrm{CH} : \{\mathrm{ID}_{v_j}, \mathrm{TTL}_{v_j}\}_{K_I}$, *Key pool generation for* v_j by $k_j^i = f(k_I, N_i)$
(3) *Erase* K_I.
(4) CH obtains the hop count: $N_{\mathrm{hops}} = \mathrm{TTL}_{\mathrm{CH}} - \mathrm{TTL}_{v_j} + 1$
(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**

Algorithm 2: Hop count and key information acquirement algorithm (TTL = 3).

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

3.4. The Common Key Discovery. For communication with ites neighbouring nodes, member node should establish secure link between them which needs the common keys to encrypt/decrypt messages. 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 $\{\mathrm{ID}_{\mathrm{CH}}, N_i, \mathrm{TTL}_i\}_{K_i}$, it also can receive $\{\mathrm{ID}_{\mathrm{CH}}, N_j, \mathrm{TTL}_j\}_{K_i}$, where $\mathrm{TTL}_i > \mathrm{TTL}_j$ $(i < j)$. 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 L_j which just stores the keys as follows:

$$L_j = \left\{ k_j^i \mid k_j^{\mathrm{Hops}}, \ldots, k_j^{\mathrm{TTL}_{v_j}+1} \right\}. \tag{3}$$

According to the principle of key generation, given two nodes v_i and v_j $(i < j)$, the set of common keys is S, then we have: $S = L_j \cap L_i = L_j$.

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 2.

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

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

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 nonce is the same in a cluster, which makes the same key for the cluster. (the last nonce is used to the cluster key).

3.5. 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 Algorithms 1 and 2, the last nonce in the set N will be transferred to every sensor.

3.6. 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 [15, 16], otherwise, when CH receives a certain amount of encrypted messages (more than $2^{2k/3}$, k is the length of key) [15], the keys will be no longer safe. According to the

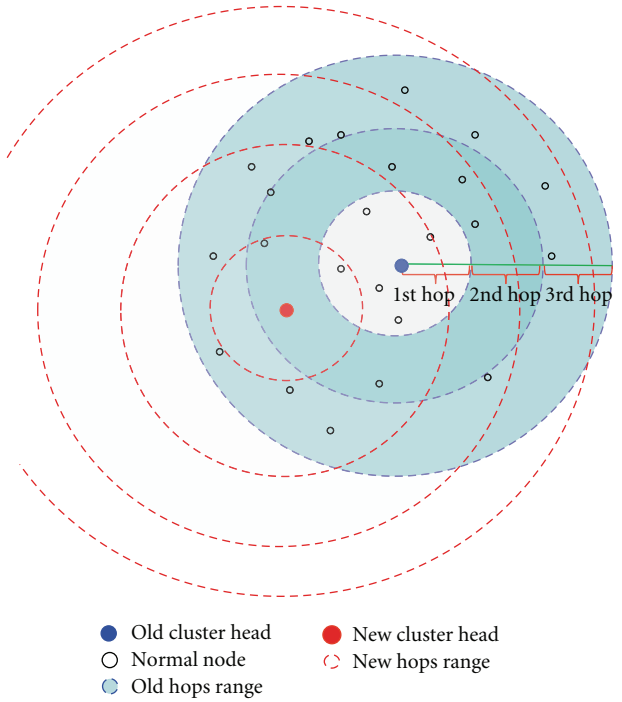

Figure 5: The reselection of cluster head.

requirements above, we should recluster after a certain phase. Assume that the process of reclustering 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 algorithm [10], it will announce itself as the cluster head and recalculate the distance from its members.

As shown in Figure 5, after reclustering, the new cluster head changes not only the relative position but also hop counts from CH to members, which make the *nonces* as well as the key things different.

4. Security Analyses

4.1. The Security Analyses. 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 the cluster formation phase, the cluster head can calculate the hop counts from the cluster head to member nodes, and then the member nodes can generate keys by the *nonces* and hops from the *beacon* message. According to the different hop counts, the cluster is divided into several security belts as shown in Figure 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.

Moreover, nodes near the cluster head 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, which

Table 3: Analysis in local key management.

Attack types	RPKH	LDK	HKMS
Selective forwarding	×	×	√
Sink-Hole attack	×	√	√
Sybil attack	√	√	√
Worm-Hole	√	√	√
HELLO flood	√	√	√
DoS	×	×	√

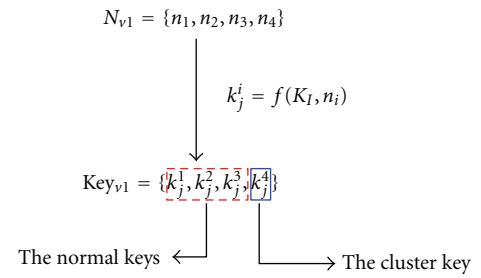

Figure 6: The cluster key in the HKMS.

makes the HKMS protocol more suitable to the collection type wireless sensor networks. The one-way security model prevents the eavesdrop attack, selective-forwarding attack DoS attack (denial-of-service) and hello flood attack, and so forth as shown in Table 3.

To communicate with members, the cluster head utilizes the last nonce as the seed of cluster key which is shared with all the sensors (including the CH) as shown in Figure 6. The cluster also can be used to rekey during the next round cluster, since the rekey process is with the redistribution of nonces. Comparing with RPKH and LDK [2, 3], the HKMS has no special requirements about the nodes, which makes it more feasibly. Also the HKMS utilizes the process of cluster formation to generate the key system without overhead, which reduces more energy consumption than previous works.

Furthermore, the key system forms during the cluster formation, which almost does not consume any energy overhead.

Furthermore, as described above, if a node can receive *beacon* message in the jth hop range, meanwhile, it also can receive *beacon* message transmitted at kth hop range, where $j < k \leq \text{TTL}$. And then, the probability can be given based on the binomial distribution as follows:

$$p_j^{\text{TTL}} = \binom{\text{TTL}}{j} \left(\frac{\text{TTL} - j + 1}{\text{TTL}}\right)^{\text{TTL}-j+1} \times \left(1 - \frac{\text{TTL} - j + 1}{\text{TTL}}\right)^{j-1}. \quad (5)$$

According to (5), with the increase of TTL, the probability also increases, that is, the common keys between two nodes are increased, which enhances the connectivity. The increase of TTL also can shrink the size of each subregion, which decreases the number of nodes who use the

Table 4: Simulation parameters.

Parameter	Value
Area size	$100 * 100$
Quantity of sensor	100
BS position	(50, 250)
Initial energy	2 J
Cluster radius	40 m
Packet size	500 Bytes
E_{elec}	50 nJ/bit
ε_{fs}	10 pJ/bit/m^2
ε_{amp}	0.0013 pJ/bit/m^4
E_{DA}	5 nJ/bit/signal
d_0	86.2 m

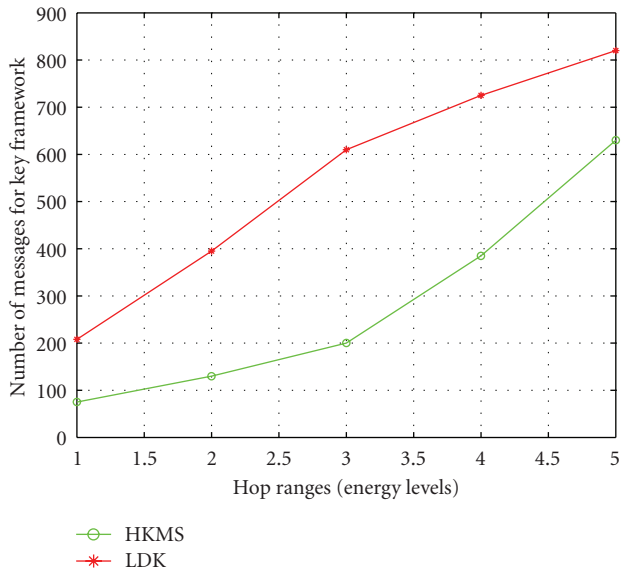

Figure 7: The comparison in energy consumption for key framework with different hop ranges (energy levels).

same communicating key and then localizes the impact of attacks. Moreover, the pair of nodes who do not have enough common keys can communicate with each other via a key path in HKMS. It also can improve the indirect network connectivity and then improve the whole network connectivity.

4.2. Simulation. In this section, we evaluate the performance of HKMS implemented in Visual C++ and MATLAB. The network scenario that we consider in simulation contains 100 nodes. According to the requirement of the HKMS, we designed a wireless sensor network simulation incorporating ECDG, essentially a multihop hierarchical sensor network [17]. The parameters for the simulations are listed in Table 4.

In Table 4, the E_{elec} is for running the transmitter or receiver circuitry; the ε_{amp} is for the transmit amplifier.

Firstly, we compare the performance of HKMS with that of LDK in energy consumption. Figure 7 shows the

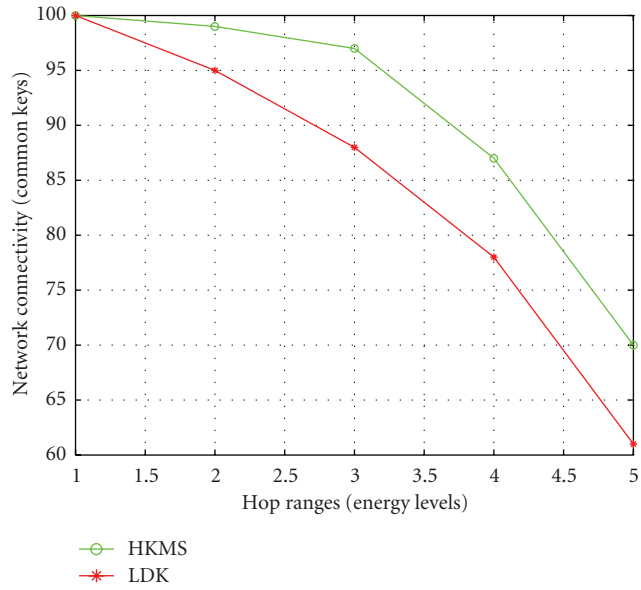

Figure 8: Comparing the performance of HKMS with that of LDK on energy consumption versus network connectivity.

comparison LDK versus HKMS in energy consumption for key framework. From Figure 7, we can see that the number of messages for key formation increased with the increase of number of hops (energy levels in LDK). For LDK, the curve looks smooth for the anchor node which has more ability to enhance the energy level to form clusters and keys. However, LDK needs transfering more messages to generate the keys. Meanwhile, with the increase of hops of HKMS, it needs more messages to be forwarded from far nodes to generate keys. Since the key things are included in the packets of ACK, the HKMS needs less message transmission. As shown in Figure 7, the HKMS uses 50% energy of LDK to form the key framework. However, due to the noise and attenuation, more hops will increase energy consumption when the hops are more than 5.

Under the same simulation environment, Figure 8 demonstrates the comparison of the connectivity of HKMS and LDK. And we can observe that more hops (energy levels) will reduce connectivity. Since the HKMS happens in one cluster, which makes it possible to communicate with each other. However, with the hop count increase, the coverage becomes bigger, which makes it difficult to forward packets. When the hops are more than 3, the QoS almost is less than 80%. For LDK, it also faces the same problem. The sensors in the radio of anchor may be not the number of the cluster. When the energy level increases, the uncertainty also increases, which reduces the connectivity.

Figure 9 shows the excepted number of keys for each member node with different TTL, which indicates that we can adjust the value of TTL to adapt to the network with the different density. Compared with LDK, HKMS has more ability to be employed for different WSNs, even in a network, there can be different size clusters because of the different TTL. In our solution, we can adjust the TTL value to average

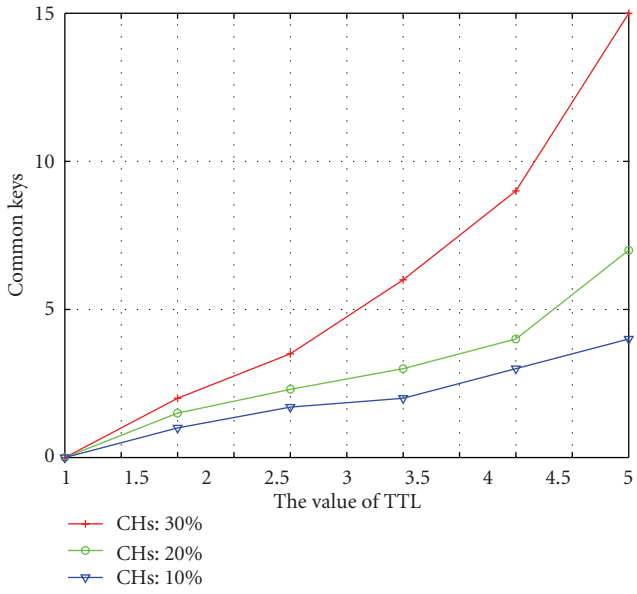

Figure 9: Excepted number of keys for each RN.

the cluster size. Figure 9 shows that the TTL value will change the density of cluster head number in the network. Here, given the TTLs are 3, 4, 5 respectively, the cluster head will be about 10%, 20%, and 30% of all nodes, respectively. From Figure 9, we can see that with the increase of TTL, the number of common keys also increases. As shown in Figure 9, when TTL is 5, the number of CHs is about 20%.

5. Conclusion and Future Work

In this paper, we propose a hierarchical key management scheme (HKMS) 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. We generate new keys in different hop ranges in a cluster. Also we present a rekey mechanism in the cluster head selection with low energy consumption. Meanwhile, HKMS can adjust the TTL to control the cluster size and the connectivity of nodes in the common keys. The simulations and security analysis show that our solution cannot only reduce the energy consumption effectively but also enhance the security level. In the future, we will focus on how to enhance security in mobile and scalable WSNs.

Acknowledgments

This work was supported by China Postdoctoral Science Foundation Funded Project (2012M510367) and this work was also supported by the following foundation: Important National Science & Technology Specific Projects of China: Next-generation broadband wireless mobile communication networks (2011ZX03005-002).

References

[1] Y. Zhou, Y. Fang, and Y. Zhang, "Securing wireless sensor networks: a survey," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 3, pp. 6–28, 2008.

[2] S. Banihashemian and A. G. Bafghi, "A new key management scheme in heterogeneous wireless sensor networks," in *Proceedings of the 12th International Conference on Advanced Communication Technology (ICACT '10)*, pp. 141–146, February 2010.

[3] F. Anjum, "Location dependent key management in sensor networks without using deployment knowledge," *Wireless Networks*, vol. 16, no. 6, pp. 1587–1600, 2010.

[4] X. Du, Y. Xiao, M. Guizani, and H. H. Chen, "An effective key management scheme for heterogeneous sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 24–34, 2007.

[5] S. Banihashemian and A. G. Bafghi, "Alternative shared key replacement in heterogeneous wireless sensor networks," in *Proceedings of the 8th Annual Conference on Communication Networks and Services Research (CNSR '10)*, pp. 174–178, May 2010.

[6] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, "MiniSec: a secure sensor network communication architecture," in *Proceedings of the 6th International Symposium on Information Processing in Sensor Networks (IPSN '07)*, pp. 479–488, April 2007.

[7] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, pp. 586–597, March 2004.

[8] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proceedings of the IEEE Symposium on Security And Privacy*, pp. 197–213, May 2003.

[9] K. Vivek, C. Narottam, and S. Surender, "A survey on clustering algorithms for heterogeneous wireless sensor networks," *International Journal of Advanced Networking and Applications*, vol. 2, no. 4, pp. 745–754, 2011.

[10] K. Ramesh and K. Somasundaram, "A comparative study of clusterhead selection algorithms in wireless sensor networks," *International Journal of Computer Science & Engineering Survey*, vol. 2, no. 4, pp. 153–164, 2011.

[11] A. Ray and D. De, "Energy efficient cluster head selection in wireless sensor network," *Recent Advances in Information Technology*, pp. 306–311, 2012.

[12] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: efficient security mechanisms for large-scale distributed sensor networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 4, pp. 500–528, 2006.

[13] A. Manjeshwar and D. P. Agrawal, "TEEN: a protocol for enhanced efficiency in wireless sensor networks," in *Proceedings of the 15th international workshop on parallel and distributed computing issues in wireless networks and mobile computing*, pp. 2009–2015, San Francisco, Calif, USA, 2001.

[14] J. S. Chen, Z. W. Hong, N. C. Wang, and S. H. Jhuang, "Efficient cluster head selection methods for wireless sensor networks," *Journal of Networks*, vol. 5, no. 8, pp. 964–970, 2010.

[15] H. N. Seyed, H. J. Amir, and D. Vanesa, "A distributed group rekeying scheme for wireless sensor networks," in *Proceedings of The 6th International Conference on Systems and Networks Communications (ICSNC '11)*, pp. 127–135, 2011.

[16] F. R. Kong and C. W. Li, "Dynamic key management scheme for wireless sensor network," *Journal of Software*, vol. 21, no. 7, pp. 1679–1691, 2010 (Chinese).

[17] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *Proceedings of the 22nd Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '03)*, pp. 1713–1723, April 2003.