Telecommun Syst (2013) 52:1377–1388
DOI 10.1007/s11235-011-9650-x

A scalable and robust hierarchical key establishment for mission-critical applications over sensor networks

Jangseong Kim · Kwangjo Kim

Published online: 14 September 2011

Abstract The previous schemes of key establishment in the wireless sensor networks may not be employed for the mission-critical application due to several limitations: lightweightness and scalability from the point of performance, vulnerabilities against node compromise and various existing attacks from the point of security. In this paper, after identifying security requirements of mission-critical applications over sensor networks, we propose a scalable and robust hierarchical key establishment scheme which enhances resilience against node capture, traffic analysis attack and acknowledgment spoofing attack. In addition, our scheme provides periodic key updates without communication costs for key transport. We verified that our scheme requires less storage, computation and communication cost compared with the previous scheme in the open literature. When AES-256 is used for symmetric encryption and one cluster consists of 50 sensor nodes, we can reduce 93.4% storage requirement and 17.2% $\sim$ 51.3% communication cost of the authentication request for the cluster. Since the reduced communication and computation costs enable the time of authentication process to be short, our scheme can support relatively fast initialization and fault recovery. Moreover, our scheme prolongs the lifetime of the wireless sensor networks.

Keywords Hierarchical key establishment · Sensor network · Mission-critical application

J. Kim (✉) · K. Kim
Department of Information and Communications Engineering, KAIST, Daejeon 305-714, Korea
e-mail: jskim.withkals@kaist.ac.kr

1 Introduction

Wireless Sensor Network (WSN) is one of the fundamental technologies for building ubiquitous computing environments. As the WSN consists of many sensor nodes with limited resources (i.e., computational power, storage and battery), it has many security vulnerabilities [1] than other networks (e.g., LAN and mesh network). To employ the sensor networks for mission-critical applications (e.g., battle field surveillance, surveillance reconnaissance, disaster relief and critical infrastructure monitoring application), we should address the following challenging problems. (1) Resilience against node capture should be enhanced as security is more important than other requirements. Since the adversary can obtain all confidential information of the compromised node, the number of the shared keys should be minimized. (2) Also, the adversary can launch various attacks during network initialization. Specially, traffic analysis may help the adversary acquire the information regarding network topology and compromise the cluster heads easily.

(3) Cryptography primitives for military application should be more lightweight. According to the implementation results in the open literature, AES-256, ECC-160 and HMAC based on SHA-160 require 15 K [3], 21 K [4] and 4 K bytes [5], respectively. Key storage of the shared keys, routing protocol, utilities, application programs and operating system should be implemented within 88 K bytes, 68.8% in the program flash memory of Mica2 sensor node [6]. However, a special-purpose program such as motion detection of any object based on image processing, which requires more than 88 K bytes, may exist.

(4) Finally, the processing time in each sensor node should be minimized. Sleeping mode, deactivation of sensor node except for radio frequency module, in the sensor network is important to extend the network lifetime. Hence, all

the processing jobs should be done in a non-sleeping phase even if the special-purpose program needs much longer processing time. Moreover, the shorter processing time supports faster network initialization, fault recovery and message delivery.

Our contribution: To address this problem, we propose a scalable and robust hierarchical key establishment scheme. While supporting periodic key update without key transport cost, providing robustness against various routing attacks and supports data aggregation reducing redundancy among sensed data within the same cluster area, compared with the previous approach [2], our scheme reduces 93.4% storage requirement and 17.2% ∼ 51.3% (or 16.5% ∼ 42.6%) communication cost for the authentication request of 50 sensor nodes in one cluster when AES-256 (or AES-128) is used for symmetric encryption. Specially, the scheme provides resilience against traffic analysis during the key establishment using the pseudonym approach.

Organization: The rest of this paper is organized as follows. Section 2 investigates the related work. Section 3 presents the assumptions and notation in this paper. Section 4 provides the detailed process of our scheme. Section 5 gives security and performance analysis. Finally, we conclude with short summary in Sect. 6.

2 Related work

We can classify the existing key establishment schemes [2, 7–10, 13] for WSN into random key pre-distribution based approach, master key based approach and trusted party based approach. In 2002, Eschenauer et al. proposed a random key pre-distribution [7]. When the whole sensor network consists of 10,000 sensor nodes, each sensor node should store 250 keys, about 6 K bytes in case of 256-bit key, in the memory to provide 99.8% network connectivity. Reduced the storage requirement, less keys in the memory, indicates a low probability to find the shared key(s). Node compromise allows the adversary to reuse the stored keys in the memory for mounting various attacks. A size of the key to be stored in each node should be increased as the network grows in size. To enhance resilience against node capture and reduce storage requirements, several methods using deployment knowledge or symmetric polynomial are introduced [8–10]. But, some problems still remain. Cooperation among the compromised sensor nodes is not considered in the analysis for resilience against node capture [11]. Random key pre-distributionapproach needs more neighbor nodes for better network connectivity. This means that the whole network size should be increased or each node should increase its transmission range which causes more frequent packet collision and communication cost [12]. The adversary can easily obtain identifiers of sensor nodes, which are useful to guess network topology information, through eavesdropping any communication.

LEAP (Localized Encryption and Authentication Protocol) is a representative scheme of master key based approach [13]. Using the shared master key and identifiers of its neighbors, each sensor node generates pairwise keys with its neighbors. Compared to the random key pre-distribution approach, LEAP provides fully connected network topology with less storage requirements, about 4 K bytes when 256-bit key is used. After generating all pairwise keys, each sensor node should erase the shared master key. But the problem is that the adversary can obtain the master key before erasing and generate all pairwise keys in the entire network. In 2005, Hartung et al. showed that anyone can get all data within 1 minute using chip-debugging method [14]. In addition, the adversary can get identifiers of sensor nodes through eavesdropping any communication.

Typical example of trusted party based approach is HIKES (HIerarchical Key Establishment Scheme) [2]. In 2007, Ibriq et al. proposed HIKES to provide robustness against well-known routing attacks while supporting the authentication and key distribution efficiently. Compared to the previous approaches, HIKES needs less storage, about 3 K bytes when 256-bit key is used. The central trust authority (or base station) transfers a part of its role for authentication and key distribution to cluster heads. As the network size increases from 1000 to 9000, according to the simulation result in [2], the energy consumption of a cluster head on key management is 3% to 20% while the cluster head in LEACH-type scheme dissipates 13% to 82% energy on key management. That's why we believe that trusted party based approach is more suitable than the other key managements. However, the adversary can reuse the stored key escrow table to guess pairwise keys of neighbors of the compromised node. Also, the adversary gets identifiers of sensor nodes. Still, HIKES requires a large amount of communications for authenticating cluster members, although it aggregates authentication message at the cluster heads.

3 Our system model, assumptions, and notation

The previous approaches [2, 7–10, 13] may be vulnerable to node compromise. In addition, an adversary can perform traffic analysis and launch several attacks to disturb the goal of the sensor network. The mission-critical applications may suffer lack of program flash memory due to cryptographic libraries (i.e., AES-256, ECC-160 and HMAC based on SHA-160) and the shared keys. The mission-critical applications should provide fast initialization and fault recovery even if the number of the deployed sensor nodes increase. Thus, the mission-critical applications should satisfy mutual authentication, source anonymity with proper accountability, confidentiality, integrity, resilience against node compromise, resilience against various existing routing attacks, fast initialization, lightweightness, scalability, and fast fault recovery.

3.1 Our system model

Figure 1 shows our system model. In this model, the sensor network consists of a base station, several gateways, multiple cluster heads and many sensor nodes. A sensor node, having a battery power, gathers the nearby interesting event (i.e., environmental information and living human in disaster area) and sends the information to a cluster head. Then, the cluster head aggregates the received information and forwards it to the base station via a gateway. Since the sensor nodes in the same cluster report very similar data compared with other nodes in the different cluster, data aggregation technique is required to extend the lifetime of the sensor network. Also, the base station, which has more computation of power, battery and storage resource compared to the other entities, sends its query or response to the cluster head and sensor node via the gateway. By introducing gateway, we can reduce energy consumption of the intermediate nodes between the target cluster head and base station. In addition, we can reduce transmission delay and packet loss due to congestion close to the base station. That's why many prototype systems support gateway [16, 17]. Hence, our system model can increase lifetime of the sensor network by reducing the number of packet retransmissions. We assume that the channel between the gateway and base station is secure.

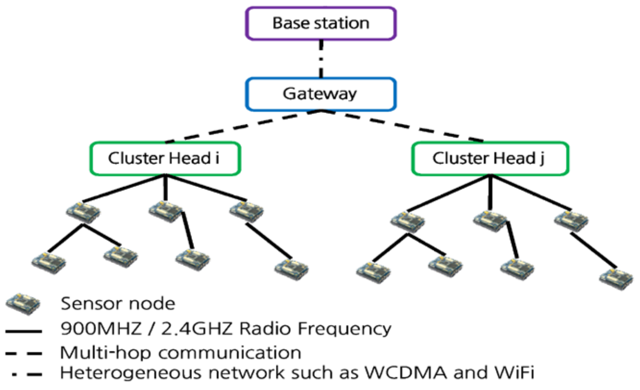


Fig. 1 System model

3.2 Assumptions and notation

In this paper, we assume that each sensor node computes HMAC of a to-be-sent packet using the preloaded master key, K_{Init} and appends the HMAC to the packet. When each sensor node receives a message (i.e., periodic or event reporting), it verifies the received message by performing HMAC with K_{Init}. If there is any modification, the node drops the received message. Through this approach, our scheme can support message integrity.

Also, each cluster head performs power adaptation for delivering a message directly to its neighbor cluster head via omnidirectional or directional antenna.

We assume that maximum hop distance between a cluster head and its member node is 2, optimal for maximum clustering effect [18].

Finally, network topology is established using K_{Init} during network initialization. The notations used throughout this paper are illustrated in Table 1.

The base station keeps initial credential C_u^0, current credential, C_u^i, a current session key, $K_{SN_u,BS}$, a selected random number j, a nonce R_u, and S in its own database per

Table 1 Notations

$BS/CH/SN$	Base station / Cluster Head / Sensor Node
$Credential$	A ticket for entity authentication
$token$	An authentication request message
n	Key update frequency, which is predetermined by the base station before node deployment.
x	A number of sensor nodes in one cluster
$K_{A,B}$	Shared secret key between entities A and B
SN_A	A sensor node having an identity 'A'
S	A set of selected numbers where $\mid S \mid$ should be larger than $2n$
C^i or $C_A^i, i = 0, 1, \ldots$	A series of authorized credentials generated by entity A
j^i or $j_A^i, i = 1, 2, \ldots$	A series of a user's number selections
$m_1 \| m_2$	Concatenation of two messages m_1 and m_2
$E\{m, K_A\}$	A message m is encrypted by a symmetric key K_A
$H(m)$	A hashed value of message m using a hash function (i.e., SHA-160)
$HMAC(m, K_A)$	HMAC operation with message m and K_A
R^i or $R_A^i, i = 1, 2, \ldots$	A series of nonces generated by entity A, which is usually a 48-bit pseudo random number

each sensor node u. Whenever a sensor node u conducts node authentication, it selects one random number j from 0 to $2n-1$ only if the j-th value of S is 0. Otherwise, the node reselects another random number j'. After encrypting j and R with the shared key $K_{SN_u,BS}$, the node sends the encryption result with its current credential C^i. Only if the node receives the proper acknowledgment from the base station, the node updates the stored S by flipping j-th value of S.

4 Our scheme

To satisfy the security and performance requirement, we adopt the following approaches. The base station authenticates all sensor nodes in the network to prevent their misbehavior and enhance resilience against node compromise. In order to reduce the advantage of node compromise, each sensor node stores and keeps only one key for authenticating itself to the base station. Although our approach needs additional communication costs compared to the previous approaches [7–10, 13], the gateway in our system model can reduce the communication costs. In addition, our approach reduces the computation and communication cost for adding a new node and removing a node. When the new node joins or leaves the sensor network, the existing approaches require the existing nodes to update the shared pairwise keys with their neighbors. Our approach enforces the existing nodes to update the shared cluster key only if some nodes in the same cluster leave the network.

We employ pseudonym and a set of the selected numbers to provide source anonymity with proper accountability. Source anonymity with proper accountability prevents the adversary from obtaining network topology information and launching a replay attack.

Each sensor node does not generate the shared keys for secure communication between the sensor node and its one-hop neighbors, but has one global key used to verify message integrity. In our approach, the adversary can only insert additional message if the neighbor node is compromised. As a result, this approach restricts the activities of the adversary. Moreover, this approach reduces the number of the shared keys with neighbors so that the processing time of each node can be reduced.

Our key establishment consists of three phases; *initialization phase*, *key establishment phase* and *update phase*. We explain each phase in detail.

4.1 Initialization phase

Each sensor node stores S, n, C^0, j^1, R^1_{BS}, and K_{Init} which are randomly generated and only shared with the base station. Then, each node computes $C^1 = H(C^0||j^1\ ||R^1)$ and generates one-time session key $K_{SN_u,BS} = H(C^0_u||C^1_u)$ where u is a node identifier.

4.2 Key establishment phase

Key establishment phase consists of three stages; gathering stage, entity authentication stage and key distribution stage.

4.2.1 Gathering stage

Each member node u, which has 2-hop distance from its cluster head, generates as the following procedure:

1. Generate a fresh nonce R^{i+1} and select $0 \le j^{i+1} \le 2n-1$ until the j^{i+1}-th value of S is 0
2. Compute own $TOKEN_u = C^i_u||E\{j^i_u||R^i_u\ ||j^{i+1}_u||R^{i+1}_u, K_{SN_u,BS}\}$ where $C^i = H(C^0||j^i||R^i)$ and $K_{SN_u,BS} = H(C^0_u||C^i_u)$

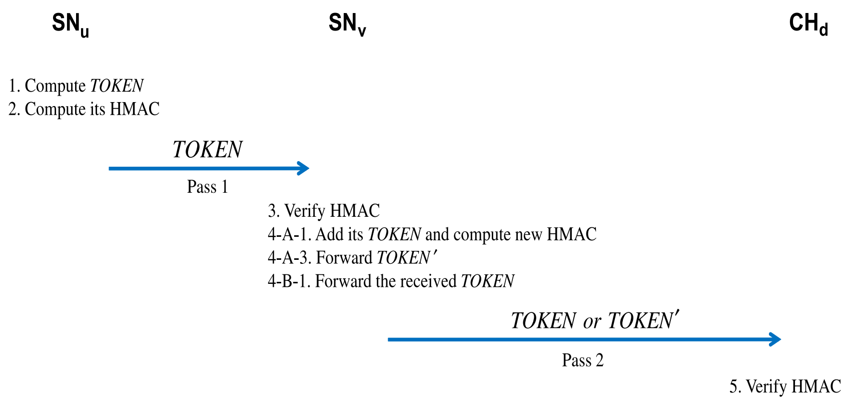
After computing its $TOKEN_u$, the node sends the token to the neighbor node v, which is a 1-hop neighbor of the cluster head. When the neighbor node v does not authenticate itself to the base station, the node computes $TOKEN_v$ and $TOKEN'$ by appending the computed token to the received message. The computation process of $TOKEN_v$ is similar to the above procedure. Then, the node v sends $TOKEN'$ to the cluster head. If the node v has already sent its own token, the node forwards the received message to its cluster head after checking message integrity. Note that the neighbor node v stores the credential, C^i_u, in the received message to its routing table. The node u can check misbehavior of the node v by overhearing the token message of the node in part of the modified authentication token. Figure 2 illustrates the gathering stage.

4.2.2 Entity authentication stage

Each cluster head sends the authentication token list, *REQ*, to the base station. Since *REQ* is ordered by the receiving sequence of the authentication tokens from member nodes, the cluster head can distribute the response message of the received token to each sensor node. The intermediate cluster heads on the path to the base station will store C^i_{CH} to their routing table and forward *REQ* to their neighbor cluster heads until the gateway receives *REQ*. Then, the gateway forwards the received message to the base station through the heterogeneous networks such as WCDMA and WiFi.

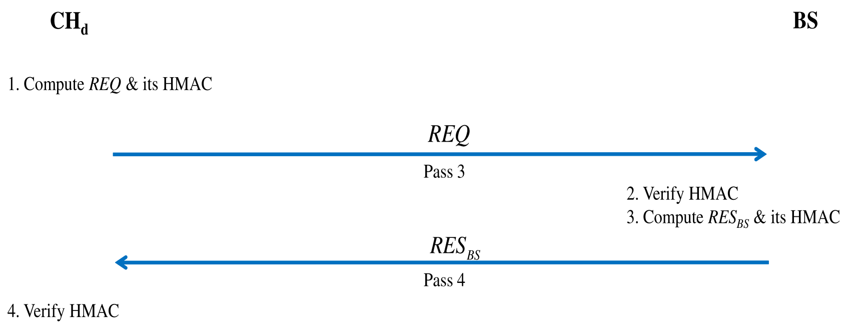
After receiving *REQ*, the base station performs the following procedures:

1. Search $K_{SN_w,BS}$ using C^i_w in each authentication token.
2. Verify the received token.
   (a) Check whether the stored R^i and j^i are the same as the received one.
   (b) Check whether j^{i+1}-th value of the stored S is 0.



$$TOKEN = C_u^i \parallel E\left\{ j_u^i \parallel R_u^i \parallel j_u^{i+1} \parallel R_u^{i+1}, K_{SN_u,BS} \right\}$$
$$TOKEN' = C_u^i \parallel E\left\{ j_u^i \parallel R_u^i \parallel j_u^{i+1} \parallel R_u^{i+1}, K_{SN_u,BS} \right\} \parallel C_v^i \parallel E\left\{ j_v^i \parallel R_v^i \parallel j_v^{i+1} \parallel R_v^{i+1}, K_{SN_v,BS} \right\}$$

Fig. 2 Gathering stage in key establishment phase



$$\begin{aligned} REQ = {} & MSG\_REQ \parallel C_{CH}^i \parallel E\left\{ j_{CH}^i \parallel R_{CH}^i \parallel j_{CH}^{i+1} \parallel R_{CH}^{i+1}, K_{CH,BS} \right\} \\ & \parallel C_1^i \parallel E\left\{ j_1^i \parallel R_1^i \parallel j_1^{i+1} \parallel R_1^{i+1}, K_{SN_1,BS} \right\} \parallel \cdots \parallel C_{x-1}^i \parallel E\left\{ j_{x-1}^i \parallel R_{x-1}^i \parallel j_{x-1}^{i+1} \parallel R_{x-1}^{i+1}, K_{SN_{x-1},BS} \right\} \end{aligned}$$
$$\begin{aligned} RES_{BS} = {} & MSG\_RES \parallel C_{CH}^i \parallel E\left\{ R_{BS}^i, K_{CH} \right\} \parallel E\left\{ K_{CH} \oplus R_{CH}^{i+1} \oplus j_{CH}^{i+1}, K_{CH,BS} \right\} \\ & \parallel E\left\{ K_{CH} \oplus R_1^{i+1} \oplus j_1^{i+1}, K_{SN_1,BS} \right\} \parallel \cdots \parallel E\left\{ K_{CH} \oplus R_{x-1}^{i+1} \oplus j_{x-1}^{i+1}, K_{SN_{x-1},BS} \right\} \end{aligned}$$

Fig. 3 Entity authentication stage in key establishment

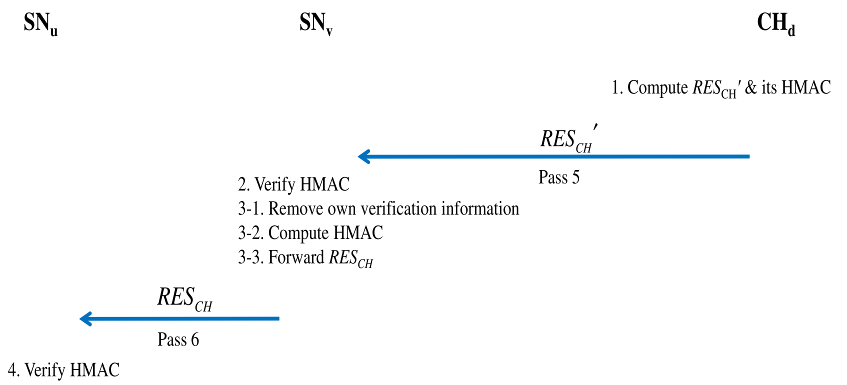
3. Compute a response message RES_w if two verification results are true.
   (a) Generate K_{CH} only once for the target cluster and compute $RES_w = E\{K_{CH} \bigoplus R^{i+1} \bigoplus j^{i+1}, K_{SN_w,BS}\}$.
   (b) Store K_{CH}, R^i and j^{i+1}.
4. Otherwise, generate an ALERT response.
5. Remove the authentication token in *REQ*.
6. Repeat (1)–(5) steps until the authentication token in *REQ* remain.

After generating all response messages for the authentication tokens in *REQ*, the base station sends a response message list, *RES*, to the target cluster head via the gateway. The intermediate cluster heads, stored C_{CH}^i in their routing table, broadcast RES_{BS} to their neighbor cluster heads until the message is delivered to the final destination. Then, the target cluster head decrypts the message, obtains K_{CH} and verifies the obtained key by comparing the obtained value R_{BS}^i with the stored R_{BS}^i. If the comparison result is not the same, the cluster re-sends *REQ* to the base station. Figure 3 depicts the entity authentication stage.

Since j^i and R^i are only known to the base station, the base station can generate C^i, decrypt $TOKEN_w$ and check whether the node w is legitimate. To support proper accountability, the base station verifies S, the set of selected number shared with each sensor node, by checking j^{i+1}-th value of the stored S.

4.2.3 Key distribution stage

Each cluster head computes a response message against each authentication token, RES_{CH} (or $RES_{CH'}$), and sends it to the member nodes. If the cluster head received $TOKEN_u$ (or $TOKEN'$), the cluster head sends RES_{CH} (or $RES_{CH'}$). Due to the characteristics of wireless communication in the WSN, all cluster members should listen the broadcasted



$$RES_{CH} = C_u^i \,||\, E\left\{R_{BS}^i, K_{CH}\right\} || \, E\left\{K_{CH} \oplus R_u^{i+1} \oplus j_u^{i+1}, K_{SN_u,BS}\right\}$$

$$RES_{CH}' = C_u^i \,||\, E\left\{R_{BS}^i, K_{CH}\right\} || \, E\left\{K_{CH} \oplus R_u^{i+1} \oplus j_u^{i+1}, K_{SN_u,BS}\right\} || \, E\left\{K_{CH} \oplus R_v^{i+1} \oplus j_v^{i+1}, K_{SN_v,BS}\right\}$$

Fig. 4 Key distribution stage in key establishment phase

message of the cluster head. After overhearing the credential in the broadcasted message, each member node can identify whether the message is sent to the node. The member node w, received the response message, performs the following procedures:

1. Classify the received message into RES_{CH} and $RES_{CH'}$ according to message size.
2. Verify its own response message.
   (a) Decrypt the authentication token and obtain K_{CH}.
   (b) Confirm K_{CH} by comparing the obtained R_{BS}^i with the stored R_{BS}^i.
   (c) Update C^{i+1}, S and $K_{SN_w,BS}$ only if the result is the same.
3. If the message is $RES_{CH'}$, compute RES_{CH} and forward it to its neighbor node.

Through the above verification of response message, the member nodes can recognize that the base station sent the response message and obtained the updated credentials of the member nodes. However, the base station updates the stored credential and shared key after receiving next authentication token from each sensor node. Without additional communication rounds, the base station cannot verify whether the sent messages are delivered to the target nodes or not. Based on these verifications, our scheme does not suffer the synchronization problem in periodic key update. Figure 4 presents the key distribution stage.

4.3 Update phase

Each sensor node is allowed to generates the authorized credential n times after node deployment. After n sessions, all nodes should update their secret information, C^0 and S, for providing anonymity with proper accountability. To support this update, we choose the base station as the update initiator. Although each sensor node can take a role of update initiator, it needs more communication cost for message delivery.

Update phases consists of update request and its distribution. In update phase, the base station sends $KEY\_UPDATE = MSG\_UPDATE||C_{CH}^i||E\{R_{BS}^{i+1}||R_{CH}^i \oplus j_{CH}^i, K_{CH,BS}\}E\{R_{BS}^{i+1}||R_1^i \oplus j_1^i, K_{SN_1,BS}\}||\cdots||E\{R_{BS}^{i+1}|| R_x^i \oplus j_x^i, K_{SN_x,BS}\}$ to the target cluster through the gateway. Then, the target cluster head decrypts first token in the received KEY_UPDATE and checks the obtained $R_{CH}^i \oplus j_{CH}^i$ with the stored value. Only if the comparison result is the same, the cluster head believes that the message is sent by the base station and updates its secret information, $S_{new} = H(S_{old}||R_{BS}^{i+1})$ and $C_{new}^0 = H(C_{old}^0||S_{new})$.

After updating its own secret information, the cluster head computes update request messages and sends them to its members. As the cluster head received an aggregated authentication token in the authentication phase, the cluster head can generate UP_REQ' and send the message to its neighbors. Then, the member node v perform the following procedures:

1. Classify the received message into $UP\_REQ = C_u^i||E\{MSG\_UPDATE||R_{BS}^{i+1}, K_{CH}\}||E\{R_{BS}^{i+1}||R_u^i \oplus j_u^i, K_{SN_u,BS}\}$ and $UP\_REQ' = UP\_REQ||E\{R_{BS}^{i+1}||R_v^i \oplus j_v^i, K_{SN_v,BS}\}$ according to the message size.
2. Verify its own update request.
   (a) After decrypting the request, extract $R_v^i \oplus j_v^i$ and R_{BS}^{i+1}.
   (b) Compare the extracted $R_v^i \oplus j_v^i$ with the stored $R_v^i \oplus j_v^i$ in its memory.
   (c) Compare the extracted R_{BS}^{i+1} with the received R_{BS}^{i+1} from its cluster head.
3. Update its secret information only if two comparison results are the same.
   (a) Compute $S_{new} = H(S_{old}||R_{BS}^{i+1})$ and $C_{new}^0 = H(C_{old}^0||S_{new})$.
   (b) Store the results in the memory.

Table 2 Security-related features

	Eschenauer et al. [7]	LEAP [13]	HIKES [2]	Ours
Mutual authentication	O	O	O	O
Periodic key update	X	X	X	O
Confidentiality	O	O	O	O
Integrity	O	O	O	O
Source anonymity	X	X	X	O
Accountability	X	X	X	O
Resilience against node capture	Low	Medium	Medium high	High
Resilience against bogus routing attack	N/A	X	O	O
Resilience against sinkhole attack	N/A	X	O	O
Resilience against sybil attack	N/A	O	O	O
Resilience against wormhole attack	N/A	X	O	O
Resilience against HELLO flood attack	N/A	O	O	O
Resilience against Ack. spoofing attack	N/A	O	X	O
Fast initialization & fault recovery	Medium high	High	Low	Medium
Scalability	Low	High	Medium	Medium high

O: Provided, X: Not provided, Ack.: Acknowledgement

4. Compute *UP_REQ* if the received message is *UP_REQ*$'$.
   (a) Remove its own update request.
   (b) Compute HMAC of the *UP_REQ* and forward *UP_REQ* to its neighbors

4.4 Improvement for key establishment phase

As the authentication of one cluster is done by the base station, the cluster head should spend more energy than the previous approaches (e.g., random key pre-distribution based approach and master key based approach). The major energy consumption of the cluster head is to forward *REQ* to the base station. As the number of the member nodes in a cluster increases, the energy consumption of the cluster head also increases linearly. Thus, our scheme may suffer frequent cluster head elections, which will reduce the expected lifetime of the sensor network.

Since the credential C_u^i of the sensor node u is used as one-time pseudonym in our scheme, we can reduce the size of each credential to the number of the deployed sensor nodes. For instance, when the network size is 60,000, 16 bits credential is enough to identify all sensor nodes in the network. The simplest way to generate the reduced size of each credential is to cut the desired length from the least significant bits of the credential. This improvement can be applied in authentication phase.

5 Analysis

In this section, we analyze the security and performance of our scheme.

5.1 Security analysis

In Table 2 we compare security-related features of our scheme with previous work.

5.1.1 *Mutual authentication with periodic key update*

Each sensor node w authenticates the base station using $K_{SN_w,BS}$ and its authorized credential. Since the credential and the shared key are only known to the base station, it persuades the base station that the token is generated by the legal sensor node. Also, each node and the base station update the shared key securely during the authentication. While each node can verify whether the base station receive necessary information for key update through key confirmation, the base station keeps the received information with the stored values until receiving the next authentication. Only if the base station can verify whether the received information is used for authenticating the node, the base station updates the stored values as the received information. Otherwise, the base station can recognize that there is an attack near to the sensor node.

5.1.2 *Confidentiality and integrity*

All communications are protected by the shared key among participants. Also, we use HMAC to provide message integrity. As a result, our scheme can achieve confidentiality and integrity requirements.

5.1.3 *Source anonymity and accountability*

By adopting pseudonym approach so that the adversary cannot identify the source of the received packet, he or she cannot obtain the network topology information and the advantage of selective packet dropping. The intermediate cluster heads, which have received *REQ*, cannot distinguish whether the message is generated by the nearby cluster head or not. Because there is no identity in *REQ* and all credentials are changed in every authentication session. However, the base station can distinguish the actual sender of the message since each sensor node generates its own authorized credential using the secret information (i.e., a set of the selected numbers and the shared key with the base station) only known to the base station and updates the credential frequently. The node w notifies the necessary information for credential update to the base station using $K_{SN_w,BS}$ and updates its secret information after key confirmation.

Using a set of the selected numbers, our scheme provides proper accountability. Because each sensor node generates its own authorized credential and updates the set of the selected numbers during n times. After n sessions, the base station sends a update request if the node is legal and does not misbehave at all. Therefore, the node can update its initial credential and set of the selected numbers. When the two or more credentials are the same, the base station will resolve this dispute using the set of the selected numbers provided by two or more sensor nodes.

5.1.4 Resilience against node capture

The major advantage of node capture is the acquisition of valid keys since the adversary can launch various attacks using those keys. In our scheme, the adversary can launch the limited attacks because he or she can get K_{Init}, K_{CH} and the shared key with the base station. When a cluster head is compromised, the adversary can change the aggregated data using K_{CH} and drop the received message (i.e., authentication request or event reporting). However, these activities can be detected by the base station and the neighboring nodes. Recall that the node u can check misbehavior of the intermediate node by overhearing the token message of the node in part of the modified authentication token. Moreover, the possibility for detection of the compromised nodes is increased with the help of the neighboring nodes since the intermediate nodes (or compromised cluster head) have only two limited permissions to check message integrity (or aggregate the received message).

Although the adversary can obtain the network topology information via node capture, it is also limited in our scheme. Because routing decision is determined by the received credentials in the routing table of each node or cluster head, the attacker can acquire no information from the credentials.

Also, the secret information for node authentication (i.e., S, j^i, R^i, R^t_{BS}, C^i and K_{CH}) are generated after node deployment. By storing the information in the volatile memory and blocking battery power when node mobility is detected, we can prevent the adversary from obtaining this information of the compromised node. The only information, which can be obtained by the adversary, is K_{Init}, C^0 and R^1_{BS}. Recall that S is updated per every authentication phase.

Finally, object detection hardware or software in military scenarios can enhance resilience against node capture. Periodic observation about the nearby environment can discover any physical approach to the sensor nodes. Then, the sensor nodes verify what it is and delete secret information except the authorized approach.

5.1.5 Resilience against various existing attacks

There are two types of attackers: outsider attackers and insider attackers. In case of outsider attack, it is relatively easy to detect the attack. Since the attacking message is generated without K_{Int}, the message fails to pass message integrity check. Now, we consider various existing routing attacks from the insider. In case of sinkhole attack, the attacker should compromise a cluster head or 1-hop neighbor v of the cluster head to establish a sinkhole. When the compromised node discards the authentication token or add several fake tokens, these activities can be detected by its neighbors or the base station. Recall that the node u can check misbehavior of the node v by overhearing the token message of the node in part of $TOKEN'$ and by performing key confirmation in authentication phase. Also, the base station can identify any insertion of fake tokens due to the increased cluster members and non-mobility. After node deployment and first authentication of the target cluster, the base station is able to bound the number of the cluster members. In addition, as node mobility is not allowed in most application scenarios, the base station can distinguish whether the adversary inserts the token through the wormhole attack. Therefore, sinkhole attack and wormhole attack are infeasible.

If the attacker launches wormhole, sybil and HELLO flood attacks, which are based on the message forwarding of legal sensor nodes in the different cluster and obtained keys from node capture, the base station identifies these attacks because of non-mobility of any sensor nodes in most applications. However, the message should be delivered to the base station. In addition, each sensor node can identify its neighbors via the received credentials during authentication phase. When the attacker launches the sybil attack, the base station recognizes that the node having the shared key with the base station is compromised due to node mobility. In case of wormhole attack, we have already discussed how to prevent this attack. Since network topology is not changed by broadcasting a HELLO packet but changing a cluster head, HELLO flood attack is also not practical. Hence, wormhole, sybil and HELLO flood attacks are infeasible.

Table 3 Computational overhead

	HIKES [2]			Ours		
	Hash operation	Symm. key operation	Nonce generation	Hash operation	Symm. key operation	Nonce generation
Pass 1	0	3	2	2	1	1
Pass 2	0	3	1	2[a] (or 4[b])	1	1
Pass 3	0	3	1	4	1	1
Pass 4	0	$x+4$	1	$2x+8$	$x+1$	0
Pass 5	1	7	0	6	1	0
Pass 6	0	4	1	4	2	0

[a]No 2-hop node exists

[b]2-hop node exists

x: The number of sensor nodes in one cluster

Acknowledgment spoofing attack can be detected since all messages from the base station contain verification information and are encrypted using the shared key with each sensor node. However, selective forwarding attack is possible in our scheme. Nevertheless, the effect is localized within a cluster (or its neighbors) if a cluster head (or a sensor node) is compromised. Compared to the previous work, our scheme provides difficulty in identifying that the packet should be dropped. Also, frequent packet drop increase a possibility being detected.

5.1.6 *Resilience against de-synchronization attack*

Since our scheme provides periodic key update, the adversary may launch de-synchronization attack using the previous credential. Although the credential is exposed to anyone, the attacker cannot generate the shared key with the base station unless compromising the target node. The base station keeps the received C^{i+1}, j^i and R^i of authorized nodes until receiving next authentication token. Only if it verifies that C^{i+1}, j^i and R^i are used in the next authentication token, it stores the received information as the authorized one. Also each sensor nodes only updates its own credential if key confirmation is successful. Thus, it is infeasible in our scheme.

5.1.7 *Fast initialization and fault recovery*

As we compared the performance of our scheme with the previous scheme in [2], our scheme reduces communication cost and average processing time in single nodes. The detailed discussion is covered in Sect. 5.2.3. Thus, our scheme can support relatively faster initialization after node deployment and faster recovery after fault occurrence on a cluster head. These features are important since mission-critical applications (e.g., military applications) need seamless service with efficient, fault-tolerance and real time delivery.

5.2 Performance

5.2.1 *Storage overhead*

To compare our scheme with the previous work [2], we assume that $|n|$, $|H(m)|$, $|K_A|$, $|C^i|$, the number of one cluster member and identifier size are 80, 20 bytes, 32 bytes, 2 bytes, 50 and 16 bits, respectively. In our scheme, all sensor nodes need to store their own secret information (i.e., K_{Init}, C^0, S, j^i, R^i, R^t_{BS}, and K_{CH}. Also, each cluster head should remember the received credential of its member nodes. While each cluster head needs 99 bytes for its own secret information and 100 bytes for received credential list, the member nodes except their cluster head needs 99 bytes for its own secret information. While the previous work [2] requires 3 K bytes, our scheme only requires 199 bytes. Thus, our scheme reduces 93.4% storage requirement.

5.2.2 *Computational overhead*

We compare the computational overhead of our scheme with HIKES [2]. Table 3 illustrates a comparison result of the computational overhead during authentication phase. To estimate processing time in the node, we compare CPU cycles between hash operation and symmetric key encryption using energy per instruction cycle. Since AES-128 consumes two times more energy than SHA-160 in MICA2 platform [19], AES needs more CPU cycle than SHA. Also, our scheme can reduce key setup time since the scheme in HIKES [2] needs more symmetric key encryptions with different keys and key setup operation in AES needs twenty times more CPU cycles than encryption [3]. Thus, our scheme needs less processing time in the node then the previous scheme [2]. Although the base station in our scheme requires more computational overhead, the burden of the base station is not critical issue due to the abundant resources such as (i.e., computation, battery and storage) of the base station.

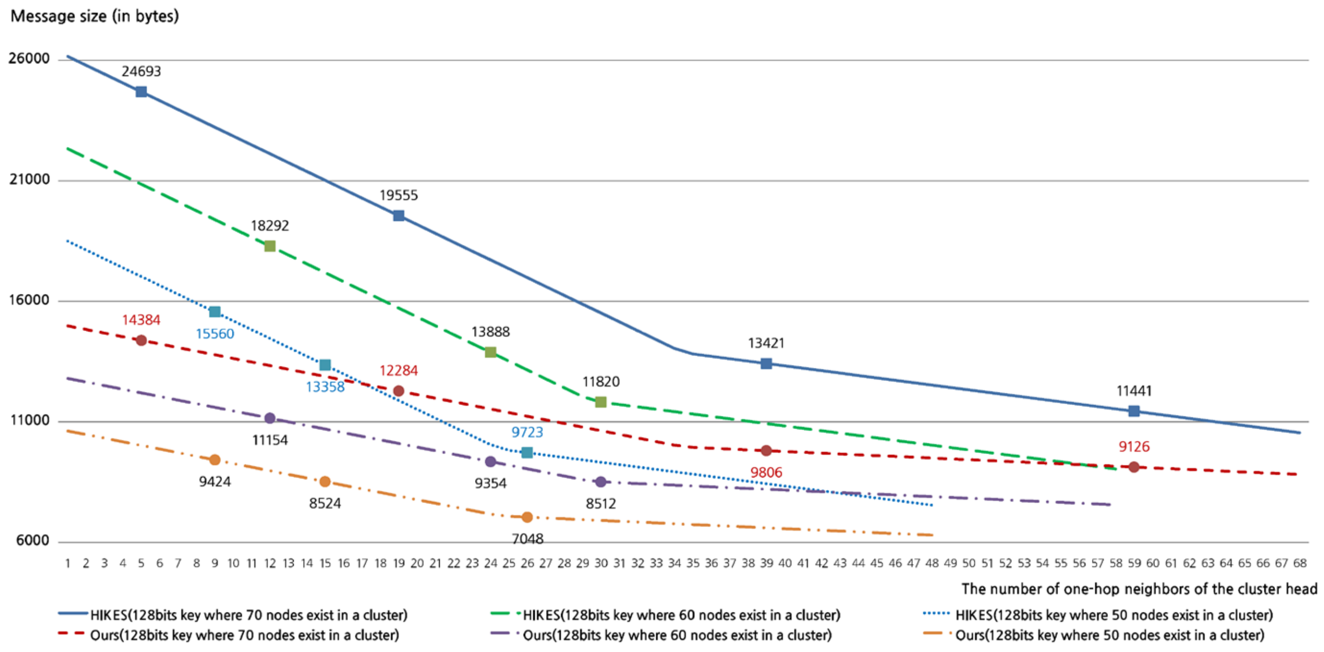


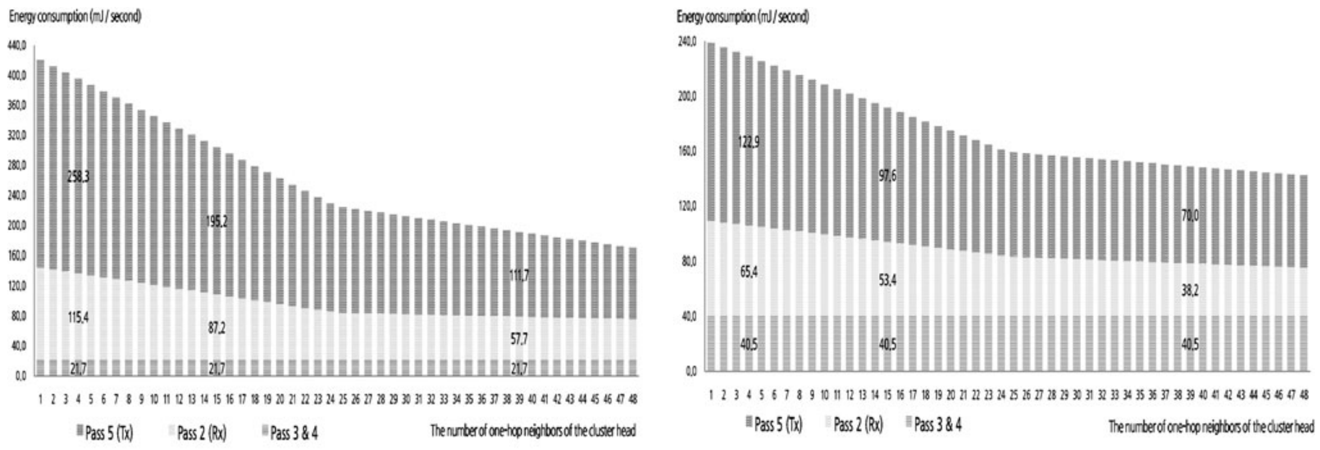
Fig. 5 Communication cost comparison of authentication process in the single cluster when AES-128 is used and the cluster consists of 50, 60 and 70 nodes

5.2.3 *Communication overhead*

We assume that HIKES [2] adopts the enhancement idea in the paper to reduce the message size during pass 5. In addition, we assume that the identifier of a sensor node is 20 bits and the size of C_u^i is 16 bits. To compare communication overhead with the previous work [2], we generate our experimental topology by distributing the member nodes having 2-hop distance in the cluster over 1-hop neighbor equally. Then, we compute the communication cost of our scheme and HIKES for authenticating all sensor nodes and a cluster head in the same cluster by adjusting the number of 1-hop neighbor of the cluster head. Note that we do not consider that communication cost of medium access control protocol in wireless sensor network. Figure 5 shows the comparison result. As the number of one-hop neighbors in the same cluster increases, the communication cost of authentication process decreases. Because the parent node can add its own authentication request to the received authentication request of a sibling node. When AES-128 is used, the number of one cluster member is 50 and the number of one-hop neighbors of the cluster head is 15, our scheme requires 8,524 bytes to authenticate all sensor nodes and one cluster head in the same cluster. However, HIKES needs 4835 bytes more. In addition, our scheme does not need to consider the communication overhead for a routing protocol while HIKES requires additional messages to support the routing protocol. In our scheme, the credential attached to the each message is used to determine the necessary routing decision. Thus, our scheme is more lightweight than the previous scheme [2].

To analysis the energy consumption of the cluster head during authentication phase, we apply the analytical model [15] proposed by Polastre et al. in 2004. Because Polastre et al. proposed the model for a real world monitoring application and validated the model by performing several microbenchmarks. Since we only focus on the energy consumption of the cluster head during authentication phase, we revise the energy consumed by transmitting, $E_{tx} = |m| \times t_{txb} \times c_{txb} \times V$ where $|m|$ is a length of message m, t_{txb} is time for transmitting 1 byte, c_{txb} is current consumption for transmitting 1 byte, and V is voltage. Also, we rewrite the energy consumed by receiving, $E_{rx} = |m| \times t_{rxb} \times c_{rxb} \times V$, where z is the number of received messages from its neighbors, t_{rxb} is time for receiving 1 byte, and c_{rxb} is current consumption for receiving 1 byte. When we use Mica2 mote, t_{txb}, c_{txb}, t_{rxb}, c_{rxb} and voltage are $416E$–6(s), 20 mA, $416E$–6(s), 15 mA and 3, respectively.

Using the information and comparison results, we computes the energy consumption of the cluster head during authentication phase. Figure 6(a) and 6(b) show actual energy consumption of one cluster head. We reduce the energy consumption of one cluster head 16.3% $\sim$ 43.2%. Whenever REQ_{BS} (or REQ) are forwarded to the neighbor cluster head, the cluster head and neighbor cluster head should spend E_{tx} and E_{rx}, respectively. Due to our assumption that the gateway forwards these messages to the base station or cluster



(a) HIKES with AES-128 (b) Ours with AES-128

Fig. 6 The energy consumption of the cluster head in single hop when the cluster consists of 50 nodes

head, we can reduce the energy E_{tx} and E_{rx}, spent by the intermediate cluster heads between the cluster head and base station.

6 Conclusion

In this paper, we proposed a scalable and robust hierarchical key establishment scheme for mission-critical applications over sensor networks. Compared with the previous work [2], our scheme enhances not only resilience against node capture and various existing attacks, but also performance with respect to storage, computation and communication. More precisely, we reduce 93.4% storage requirement, the number of symmetric key operations in single sensor node and 17.2% $\sim$ 51.3% (or 16.5% $\sim$ 42.6%) communication cost for the authentication request of 50 sensor nodes in one cluster when AES-256 (or AES-128) is used for symmetric key encryption. Using this novel property, our scheme can support fast initialization and fault recovery. Moreover, our scheme provides source anonymity to obstruct the adversary to obtain network topology information and determine which message should be dropped. By reducing the resources required to authenticate the sensor nodes in one cluster and providing better security properties than the previous work, we obtain better scalability and robustness.

In the near future, we will extend our scheme to build secure routing protocol for military applications and analyze their performance. Note that secure routing protocol includes cluster formation, discovery of neighbor cluster, data aggregation and reporting, and topology maintenance.

References

1. Karlof, C., & Wagner, D. (2003). Secure routing in wireless sensor networks: attacks and countermeasures. *Ad Hoc Networks*, *1*, 293–315.
2. Ibriq, J., & Mahgoub, I. (2007). A hierarchical key management scheme for wireless sensor networks. In *21st international conference on advanced networking and applications (AINA)*, May 21–23, 2007, Niagara Falls, Canada (pp. 210–219).
3. Law, Y. W., Doumen, J., & Hartel, P. (2006). Survey and benchmark of block ciphers for wireless sensor networks. *ACM Transactions on Sensor Networks*, *2*(1), 65–93.
4. Liu, A., & Ning, P. (2008). TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks. In *Information processing in sensor networks (IPSN '08)*, 22–24 April, 2008 (pp. 245–256).
5. Lee, H. R., Choi, Y. J., & Kim, H. W. (2005). Implementation of TinyHash based on Hash algorithm for sensor network. In *Proceedings of world academy of science, engineering and technology*, Dec. 2005 (vol. 10, pp. 135–139).
6. Crossbow datasheet on Mica2 (2010). http://www.xbow.com/products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf, April, 2010.
7. Eschenauer, L., & Gligor, V. D. (2002). A key management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on computer and communications security (CCS)*, Washington, DC, USA, November 18–22, 2002 (pp. 41–47).
8. Du, W., Deng, J., Han, Y. S., & Varshney, P. K. (2003). A pairwise key predistribution scheme for wireless sensor networks. In *Proceedings of the 10th ACM conference on CCS*, Washington, DC, USA, October 27–31, 2003 (pp. 42–51).
9. Liu, D., & Ning, P. (2003). Establishing pairwise keys in distributed sensor networks. In *Proceedings of the 10th ACM conference on CCS*, Washington DC, USA, October 27–31, 2003 (pp. 42–61).
10. Du, W., Deng, J., Han, Y. S., Chen, S., & Varshney, P. K. (2004). A key management scheme for wireless sensor networks using deployment knowledge. In *Proceedings of the 23rd conference on the IEEE INFOCOM*, Hong Kong, China, March 7–11, 2004.
11. Moore, T. (2006). A collusion attack on pairwise key predistribution schemes for distributed sensor networks. In *Proceedings*

of the fourth annual IEEE international conference on pervasive computing and communications workshops, March 13–17, 2006 (pp. 251–255).
12. Silva, R. M. S., Pereira, N. S. A., & Nunes, M. S. (2007). Applicability drawbacks of probabilistic key management schemes for real world applications of wireless sensor networks. In *Proceedings of the third international conference on wireless and mobile communications*, March 4–9, 2007 (p. 51).
13. Zhu, S., Setia, S., & Jajodia, S. (2003). LEAP: efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 10th annual conference on CCS*, Washington DC, USA, October 27–31, 2003 (pp. 62–72).
14. Hartung, C., Balasalle, J., & Han, R. (2005). *Node compromise in sensor networks: the need for secure systems* (Technical Report CU-CS-990-05) January, 2005.
15. Polastre, J., Hill, J., & Culler, D. (2004). Versatile low power media access for wireless sensor networks. In *Proceedings of the 2nd international conference on embedded networked sensor systems*, Baltimore, MD, USA, November 3–5, 2004 (pp. 95–107).
16. Beckwith, R., Teibel, D., & Bowen, P. (2004). Pervasive computing and proactive agriculture. In *Adjunct proceedings PERVASIVE computing and proactive agriculture*, Vienna, Austria, April 2004.
17. Kappler, C., & Riegel, G. (2004). A real-world, simple wireless sensor network for monitoring electrical energy consumption. In *Proceedings of first european workshop on wireless sensor networks*, Berlin, Germany, January 2004 (pp. 339–352).
18. Vlajic, N., & Xia, D. (2006). Wireless sensor networks: to cluster or not to cluster? In *Proceedings of the 2006 international symposium on a world of wireless, mobile and multimedia networks*, Niagara-Falls, Buffalo-NY, June 25–29, 2006
19. Chang, C. C., Muftic, S., & Nagel, D. J. (2007). Measurement of energy costs of security in wireless sensor nodes. In *Proceedings of 16th international conference on computer communications and networks*, Washington DC, USA, August 13–16, 2007 (pp. 95–102).

Jangseong Kim received the B.S. degree in Computer Engineering from Kyungpook University, Korea. He is presently Ph.D. candidate in Information and Communications Engineering, Korea Advanced Institute of Science and Technology, Korea. His interests include security and privacy issue in ubiquitous computing environment, wireless sensor network security, and VANET security.

Kwangjo Kim received the B.S. and M.S. degrees of Electronic Engineering in Yonsei University, Korea, and Ph.D. of Div. of Electrical and Computer Engineering in Yokohama National University, Japan. Currently he is a professor at Computer Science Department in Korea Advanced Institute of Science and Technology, Korea. He served the president of Korean Institute on Information Security and Cryptology (KIISC) in 2009 and Board Member of IACR from 2000 to 2004. His research interests include the theory of cryptology and information security and their practice.