A Secure Key Agreement Scheme in Low-energy Wireless Sensor Networks

Taeyeon Kim[1], Gicheol Wang[2], and Gihwan Cho[3]

[1] Dept. of Computer Science and Information Communications,
Seonam University, Namwon, Jeonbuk, 590-711, Republic of Korea
kcopper7@hanmail.net
[2] CAIIT, Chonbuk National University, Jeonju,
Jeonbuk, 561-756, Republic of Korea
gcwang@dcs.chonbuk.ac.kr
[3] CAIIT, Division of EIE, Chonbuk National University,
Jeonju, Jeonbuk, 561-756, Republic of Korea
ghcho@dcs.chonbuk.ac.kr

Abstract. Ubiquitous computing environment provides users with information access anytime and anywhere. In particular, sensor networks must be broadly deployed in real world and utilized to ensure the safety of the human life. In cryptography aspect, key agreement protocol is very important element to exchange messages safely between sensor nodes. This concern originates from the fact that sensor nodes are highly vulnerable to faults, energy depletions, and security attacks. The open problems are how to verify the identity of communicating nodes, how to set up a session key between communicating nodes, and how to minimize any information about the keys disclosed to the other side of key agreement. To solve above problems, we propose a secure key agreement scheme for low-energy sensor networks. Our scheme is based on the MRS scheme and enhances the security by hiding unshared keys and the number of shared keys. Besides, it resolves the weak points in encryption mechanism of MRS by employing multiple random numbers. Performance and security analyses have proven that our scheme is suitable for sensor networks in terms of availability and security aspects.

1 Introduction

Wireless Sensor Network (WSN) is known as a new communication infrastructure for the future computing era. Usually it requires no centralized center or fixed network infrastructure, and can be deployed quickly and inexpensively as needed. The sensor nodes collaborate to collect, process, analysis, and disseminate the sensed data in hostile environments, eventually in order to provide users information access anytime and anywhere. However, the individual sensor node suffers from limited resources, battery, memory, processor, network bandwidth, and so on.

Many studies focus on the routing, fault recovery, energy-efficient communication, and security issues for WSN [1]. Due to the resource scarcity, these concerns are originated from the fact that nodes are highly vulnerable to faults of sensor nodes, energy depletions, and security attacks. Therefore, it is important to invent a communication protocol which satisfies the security requirements as well as the energy saving of nodes. For example, malicious nodes can easily listen to the traffic, impersonate one of the nodes, or provide misleading information to other nodes intentionally. Therefore, the communication between nodes in hostile environments should be authenticated and encrypted.

It is natural that common key management techniques using asymmetric cryptographic algorithms are not appropriate for WSN due to limited resources; it is natural to use symmetric cryptographic algorithms in WSNs because they are relatively fast and induce low cost for cryptographic processes.

Almost all of key pre-distribution schemes [2-6] assumed that a random graph *G(n, P)* is a graph of *n* nodes for which the probability that a link exists between two nodes is *P*. And each one fully trusts the other side during the key pre-distribution. Without a doubt, nodes disclose their key information to neighboring nodes which held a random subset of keys of the key pool. At any rate, the existing protocols, which are based on random key pre-distribution, can not perfectly satisfy requirement of the key management due to some drawbacks. The open problems are how to authenticate the key information of communicating nodes, how to securely set up a session key between communicating nodes, and how to minimize the amount of disclosed information about the keys to the other side.

To solve the first problem, almost all the schemes so far rely on three phases as followings: key pre-distribution, shared-key discovery and path-key establishment. But the second and the third problems are big challenges that are not yet solved.

In the existing schemes, if two nodes share at least one key, they consider each other to be worthy of confidence and generate a session key for further communication between them. But if an adversary accidentally generates the same key(s) that any key(s) in the key pool to impersonate one of legitimate nodes, he can get a session key after completing shared-key discovery phase.

Recently, Chan [7] proposed a key agreement scheme where each node can find keys shared with a communicating node without revealing the unshared keys. However, in this scheme, some security problems are discovered. For example, before completing shared-key discovery phase, a malicious node can know how many keys in the key chain held by itself are shared with the other side without receiving the other side's response. Also, it can guess some of the keys held by the other side due to the weakness of encrypted values received from the other side.

In security aspect, disclosure of a secret key itself causes a lot of threats to legitimate nodes. This paper proposes a novel key agreement scheme which resolves the second and third problem caused during the key discovery phase in WSNs. It includes some methods by which each node authenticates the secret keys received from the other side, prevents the disclosure of unshared keys as well as the exposure of number of shared keys, and strengthens a cryptographic algorithm.

The rest of the paper is organized as follows. In section 2 and 3, we present the modified Rivest's scheme and the secure key agreement scheme for WSNs. In section 4, the performance results and security analyses are described. Finally, section 5 concludes this paper.

2. Modified Rivest's Scheme (MRS)

In the rest of this paper, we use the following notation.

- n : size of network
- z, s, m : size of the key space, the key pool and the key chain respectively
- SK : session key generated between authorized nodes
- KA, KB : secret key of node A and B
- $E^K(M),\ D^K(M)$: message M encrypted and decrypted with key K
- $h()$: one-way hash function
- p, q : prime numbers
- r (or r_i), s (or s_i) : random numbers, $0 \le i < m$.

Before we describe our scheme, we review the modified Rivest's scheme which is based on a scheme in [8]. A detailed scheme is described in [7]. The key pre-distribution phase ensures that each node is assigned a random subset of keys, *m*, from a key pool before deployment. And in shared-key discovery phase, each node finds the keys shared with the other side node. It does not disclose any information about the keys that the other side does not have. The algorithm for the encryption and decryption of a message and the shared-key discovery phase are performed as follows.

[Encryption] To encrypt a message $M \in Z_n$, the following steps are performed:

- Break down M into a number of arbitrary pieces - $(a_1, a_2, ..., a_k)$ in such a way that $M = \sum_{i=1}^{k} a_i \bmod (p \times q)$.
- Randomly choose $r_i < p$ and $s_i < q$, $\forall_i \in [1, k]$ (which are kept secret).
- Apply the encryption function like the following.

$$E_{p,q,r_i,s_i}(M) = ((a_1 r_1 \bmod p, a_1 s_1 \bmod q), ..., (a_k r_k \bmod p, a_k s_k \bmod q))$$
$$= ((x_1, y_1), (x_2, y_2), ..., (x_k, y_k))$$

[Decryption] Given $X = ((x_1, y_1), (x_2, y_2), ..., (x_k, y_k))$, the decryption steps are as follows:

- $E_{p,q,r_i,s_i}(M) = ((x_1 r_1^{-1} \bmod p, y_1 s_1^{-1} \bmod q), ..., (x_k r_k^{-1} \bmod p, y_k s_k^{-1} \bmod q))$
- Use Chinese Remainder Theorem to find $(a_1, a_2, ..., a_k \pmod{(p \times q)})$.
- Sum up a_i's to recover M . In such a way that $M = \sum_{i=1}^{k} a_i \bmod (p \times q)$

MRS ensures which the componentwise addition and multiplication (mod (p×q)) of the ciphertexts are the same as the encrypted values of the addition and multiplication of the corresponding plaintexts, and a value should have a number of different possible representations in the ciphertext domain.

When MRS scheme is used in the shared-key discovery phase, each node makes use of a polynomial expression. For example, suppose that Alice wants to find out the common keys with Bob, and Alice has the key set $A = \{a_1, a_2, ..., a_m\}$ and Bob has the key set $B = \{b_1, b_2, ..., b_m\}$. Alice forms a polynomial expression:

$$f_A(x) = (x - a_1)(x - a_2)...(x - a_m) = x^m + A_{m-1}x^{m-1} + ... + A_1 x + A_0 \quad (1)$$

and sends the encrypted coefficients of $f_A(x)$ to Bob.

Alice → Bob : $E^K(A_0), E^K(A_1), ..., E^K(A_{m-1})$, where K is a secret key which she has only.

Bob forms the following polynomial expression using the received message:

$$f_A^{'}(x) = x^m + E^K(A_{m-1})x^{m-1} + ... + E^K(A_1)x + E^K(A_0) \quad (2)$$

To generate a list of encrypted values (i.e., $rE^K(B_0), rE^K(B_1), ..., rE^K(B_{m-1})$), he applies his keys to the expression (2). Here, $E^K(B_i)$ is $f_A^{'}(B_i)$ and r is a random number.

Bob → Alice : $rE^K(B_0), rE^K(B_1), ..., rE^K(B_{m-1})$

She applies $D^K(rE^K(B_{i-1}))$ and can get $rf_A(b_i))$ for $1 \le i \le m$. Since she has no knowledge about r, she does not know what b_i is. But, if anything in $rf_A(b_i))$ is zero, she knows that two nodes share at least one key. Otherwise, she knows that they share no keys with each other. Also, for $rf_A(b_i)) \ne 0$, since there are $u+1$ unknowns in non-linear equations(for some $u \le m$), it would not be possible for Alice to find b_i in the information theoretic sense. When the protocol is still in process, since Bob does not know two large primes p and q, and random number r_i and s_i, he also cannot know which one in a list of encrypted values is an encrypted zero. Consequently, after the shared-key discovery, each node can find shared keys only without revealing unshared keys.

3. Secure Key Agreement Scheme

The proposed scheme leverages Eschenauer and Gligor's scheme [2] and Modified Rivest's Scheme [7]. Sensor networks consist of base stations and sensor nodes. The base station is assumed to be computationally robust and installed in a fixed and secure location. In the remainder of this paper, we make use of their algorithm as the underlying scheme. The path-key establishment phase will not be described since it is assumed that our scheme employs the same protocol as the scheme [2] proposed.

3.1 Key pre-distribution phase

In the initialization phase, the base station picks a random key pool out of the total possible key space. Also, a key information in the key pool is combined a secret key (K_i) in the key space with a one-way hash function (h_i), (K_i , h_i), $0 \le i \le z-1$. Each node randomly picks a key chain (i.e., $(K_i, h_i), i = 1, ..., m$) from the key pool before deployment. The key chain is utilized to generate a session key between two nodes during the key discovery phase. And the hash function is utilized to authenticate the secret keys (K_i). It is for the sake of decreasing the possibility that malicious nodes intentionally generate a random key chain.

3.2 Key agreement phase

3.2.1 Negotiatory Keys (NKs)

After deployment, each node needs to find whether it shares any key with its neighbors. To do this, each node generally generates *m* non-linear equations with the secret keys it carries as expression (1) and broadcasts the message containing the encrypted coefficients of $f_A(x)$.

But in this paper, we make use of negotiatory keys so as to enhance security. The generation of negotiatory keys is as follows. For example, let Alice has the key set $A = \{a_1(= a_{11} \| a_{12}), a_2(= a_{21} \| a_{22}), ..., a_m(= a_{m1} \| a_{m2})\}$, Bob has the key set $B = \{b_1(= b_{11} \| b_{12}), b_2(= b_{21} \| b_{22}), ..., b_m(= b_{m1} \| b_{m2})\}$, and $h()$ generates the value of limited length. Also, r_i and $r_i^{'}$ are random numbers where $r_{i-1} \neq r_i$ and $r_{i-1}^{'} \neq r_i^{'}$ for $1 \leq i \leq m$. We yield a negotiatory key by concatenating the first half of the key and a hash function's value, and in reverse. That is, the half of Alice's negotiatory keys consists of $\{s_{11}(= a_{11} \| h(a_{11})), s_{21}(= a_{21} \| h(a_{21})), ..., s_{m1}(= a_{m1} \| h(a_{m1}))\}$. The other half consists of $\{s_{12}(h(a_{12}) \| a_{12}), s_{22}(h(a_{22}) \| a_{22}), ..., s_{m1}(h(a_{m2}) \| a_{m2})\}$. Similarly, the first half of Bob's negotiatory keys consists of the following. $\{t_{11}(= b_{11} \| h(b_{11})), t_{21}(= b_{21} \| h(b_{21})), ..., t_{m1}(= b_{m1} \| h(b_{m1}))\}$. Also, the second half consists of $\{t_{12}(= h(b_{12}) \| b_{12}), t_{22}(= h(b_{22}) \| b_{22}), ..., t_{m2}(= h(b_{m2}) \| b_{m2})\}$.

3.2.2 Shared-key discovery

In this section, we describe the way how two nodes calculate their session key. Alice who wants to establish a session key generates negotiatory keys and adapts them to expression (3). Then she encrypts coefficients of $f_A(x)$ with her secret key (*KA*) (i.e. $E^{KA}(A_0), E^{KA}(A_1), ..., E^{KA}(A_{m-1})$) and broadcasts them to neighboring nodes. In reverse, when she receives a requesting message from the other side (i.e. Bob), she applies her negotiatory keys to expression (6) in order to generate a list of encrypted values (i.e. $r_0 f_B^{'}(s_{12}), r_1 f_B^{'}(s_{22}), ..., r_{l-1} f_B^{'}(s_{m2})$) and sends them to Bob. Bob also generates negotiatory keys and adapts them to expression (5). Then he encrypts coefficients of $f_B(x)$ with his secret key (*KB*) (i.e. $E^{KB}(B_0), E^{KB}(B_1), ..., E^{KB}(B_{m-1})$) and broadcast them to neighboring nodes. In reverse, when he receives a requesting message from other side (i.e. Alice), he applies his keys to expression (4) in order to generate a list of encrypted values (i.e. $r_{_0}^{'} f_A^{'}(t_{12}), r_{_1}^{'} f_A^{'}(t_{22}), ..., r_{m-1}^{'} f_A^{'}(t_{m2})$) and sends them to Alice.

However, two nodes decrypt $r_i^{'} f_A^{'}(t_{i1})$ and $r_i f_B^{'}(s_{i2})$ with their secret key respectively. And Alice sends an *m*-bit bitmap with 1 at bits where $D^{KA}(r_i^{'} f_A^{'}(t_{i1})) = 0$ to

Bob, and Bob sends an *m'*-bit bitmap with 1 at bits where $D^{KB}(r_i f_B^{'}(s_{i2}))=0$ to Alice. To strengthen security, each node reduces the number of bits with 1 in its bit bitmap by one-half. The detailed description is as follows.

$$f_A(x)=(x-s_{11})(x-s_{21})...(x-s_{m1})=x^m+A_{m-1}x^{m-1}+...+A_1x+A_0 \qquad (3)$$

$$f_A^{'}(x)=x^m+E^{KA}(A_{m-1})x^{m-1}+...+E^{KA}(A_1)x+E^{KA}(A_0) \qquad (4)$$

$$f_B(x)=(x-t_{12})(x-t_{22})...(x-t_{m2})=x^m+B_{m-1}x^{m-1}+...+B_1x+B_0 \qquad (5)$$

$$f_B^{'}(x)=x^m+E^{KB}(B_{m-1})x^{m-1}+...+E^{KB}(B_1)x+E^{KB}(B_0) \qquad (6)$$

1) Alice calculates encrypted coefficients of $f_A(x)$ in expression (3), $E^{KA}(A_0), E^{KA}(A_1), ..., E^{KA}(A_{m-1})$ and sends them to Bob.

2) ⓐ Bob, on receiving the encrypted coefficients, applies them to expression (4) and gets $f_A^{'}(t_{i1})$, for $1 \le i \le m$. To strengthen security, Bob chooses random numbers $r_i^{'}$ and calculates $M^{'}=r_1^{'}f_A^{'}(t_{11}), r_2^{'}f_A^{'}(t_{21}), ..., r_m^{'}f_A^{'}(t_{m1})$. Where $r_i^{'}$ are different values and nonzero.

ⓑ As the above, he calculates encrypted coefficients of $f_B(x)$ in expression (5), $E^{KB}(B_0), E^{KB}(B_1), ..., E^{KB}(B_{m-1})$.

ⓒ He sends $M^{'}$ and the encrypted coefficients to Alice.

3) ⓐ Alice decrypts $M^{'}$, $D^{KA}(r_i^{'}f_A^{'}(t_{i1}))$, and calculates an *m*-bit bitmap with 1 at bits where $D^{KA}(r_i^{'}f_A^{'}(t_{i1}))$ is 0 and 0 elsewhere. A 1 at the *i*-th bit indicates to Bob that she also has t_{i1}. To enhance security, if the number of bits with 1 in *m*-bit bitmap (i.e. *W*) is more than 1, she randomly adjusts it to *w* (=$\left\lceil \frac{W}{2} \right\rceil < m$).

ⓑ She applies the other side's encrypted coefficients to expression (6) and gets $f_B^{'}(s_{i2})$, for $1 \le i \le m$. Alice chooses random numbers r_i and calculates $M=\{r_1 f_B^{'}(s_{12}), r_2 f_B^{'}(s_{22}), ..., r_m f_B^{'}(s_{m2})$, where r_i s are different values and nonzero.

ⓒ She sends M and an *m*-bit bitmap to Bob.

4) ⓐ Bob decrypts M, $D^{KB}(r_i f_B^{'}(s_{i2}))$ and calculates an *m'*-bit bitmap with 1 at bits where $D^{KB}(r_i f_B^{'}(s_{i2}))$ is 0 and 0 elsewhere. To enhance security, if the number of bits with 1 in *m'*-bit bitmap (i.e. *W'*) is more than 1, he randomly adjusts it to *w'* (=$\left\lceil \frac{W^{'}}{2} \right\rceil < m$).

ⓑ He sends the *m'*-bit bitmap to Alice.

5) Each node generates a session key using both *m*-bit and *m'*-bit bitmap. That is, a new session key SK is generated as the hashed value of the concatenation of shared keys (i.e. $SK = h(K_1 \| K_2 \| ... \| K_{w'})$).

4. Performance and Security Analysis

4.1 Probability of sharing at least one key

An event-driven simulator has been developed to evaluate availability and security of proposed scheme. Our approach is compared with MRS proposed by Chan [7]. In our simulations, we induced two metrics to evaluate the availability and the security of the proposed scheme. One metric is the actual probability that any two neighboring nodes share at least one key during a key agreement phase. The other metric is the rate that all session keys are exposed to an attacker under the existence of one compromised node. For the sake of presentation, our scheme is hereafter referred to as SKS (Secure Key agreement Scheme).

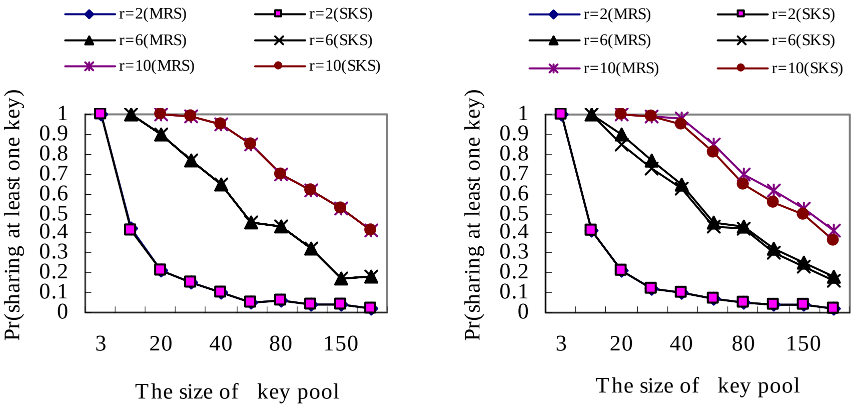
The network model for our simulation assumed as follows; ⒜ 200 nodes were randomly placed in a 100m × 100m area. ⒝ the length (r) of the key chain varied in 2, 6, and 10. ⒞ number of cases that, in key pool, the first half of key is same to others is varied in 0% and 30%.

As shown Fig. 1, as the size of key pool increases, the probability that any two neighboring nodes share at least one key also decreases. In both schemes, if the first half of the keys are not the same to others', the key sharing probability is identical (see Fig. 1(a)). However, as the cases that the first half of keys are same to others increased to 30% (See Fig. 1(b)), the key sharing probability makes a little difference between two schemes, although very little.

Next, to evaluate the security of the proposed scheme, we estimated the exposure rate of session keys during a key agreement under the existence of one compromised node by an attacker. In our simulations, the compromised nodes were randomly selected. Fig. 2 shows the session key exposure rate when a node is compromised by an attacker. As shown in Fig. 2, even though a node is compromised by an attacker, the proposed scheme is much less affected than the MRS. This is because the proposed scheme hides the keys unshared with other nodes. Therefore, the proposed scheme is robust against the compromise of nodes.

4.2 Key authentication

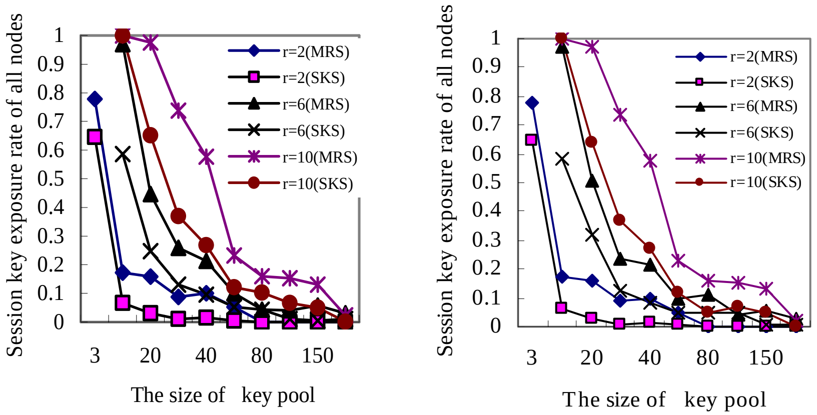
Upon and after network initialization, in order to increase the communication and computation overhead of networks, a malicious node can broadcast a random key chain falsified by itself to neighboring nodes. If any keys in the falsified key chain are in common with the other side, the attacker can establish a secure link with the legitimate node. However, since our scheme makes use of negotiatory key to provide the authentication of key information, it guarantees the key authentication. Even if an attacker luckily generates a key shared with a legitimate node, it can not generate a session key for further communication between two nodes. This is because it has no corresponding one-way hash function.



(a) Number of cases = 0%

(b) Number of cases = 30%

Fig. 1 Key sharing probability vs. key pool size (zero and 30% match case)



(a) Number of cases = 0%

(b) Number of cases = 30%

Fig. 2 Session key exposure rate vs. key pool size (zero and 30% math case)

4.3 Reducing the number of disclosing shared keys

Before completing the MRS protocol, malicious node can know how many keys of the key chain are in common with the other side without the other side's response. In the worst case, it is not difficult to guess any shared keys while the ratio of the shared keys to unshared keys is by far higher than the reverse of it. Eventually, other side may suffer from potential attacks even if the shared-key discovery is completed before finishing an attack.

Also, after finishing the shared-key discovery phase in MRS, each pair of nodes can know all the shared keys but they can not know unshared keys. However, if one node of them is compromised by adversaries later, the other side also is easy to suffer from attacks by malicious nodes. And one of the two nodes can use random messages in order to search any keys among the other side's keys. That is, to mount an exhaustive discovery for keys held by his neighboring nodes by intentionally, the node may encrypt coefficients using random secret keys and broadcast them. After a completion of key discovery process, it would be possible to know any secret keys held by his neighboring nodes.

To surmount these problems, we made use of negotiatory keys instead the secret keys that is generally used by the existing schemes and mechanism which restricts the number of disclosed shared keys. Consequently, if some nodes are captured, the probability that a session key between any two nodes is affected by malicious nodes is decreased considerably.

4.4 Preventing attack of ciphertexts

Simple encryption of a message does not absolutely assure that the message will not be revealed during or after key discovery. That is, if MRS scheme is used in the shared-key discovery phase, nodes face to another threat caused by guessing secret keys and striking weak points of encryption. After an adversary receives all the messages from anyone in the network, the node can guess any secret keys held by the other side due to a weakness of encrypted values, which received from the other side. Let's suppose that an adversary received a list of encrypted values (i.e. $rE^K(D_0), rE^K(D_1), ..., rE^K(D_{m-1})$) from any legitimate node. If two parties have any common key (i.e., c_i) in their key chains, the encrypted value(s) which corresponds to it are decrypted to zero. Otherwise all encrypted values are decrypted to nonzero. However, even if node A has no knowledge about *r*, he can obtain factorization expressions as follows:

$$D^K(rE^K(D_i)) = r \times (Y_i)^m, i = 0...m-1 \quad (7)$$

It is not easy to guess two numbers (r and Y_i) from *m* different values as the expression (7). On the other hand, it is not difficult to guess them. That is, the node can generate any keys, Y_i , that held by the other side using his secret key *K*, two large prime numbers *p* and *q*, and a random number r . If the number of $D^K(rE^K(D_i)) \neq 0$ is more than 3, it becomes easier.

Therefore, the number *r* needs to be managed deliberately. To do this, we made use of different random numbers instead of a random number. As a result, it is not easy for a malicious node to find a value c_i or any secret key held by other side because it is very difficult to guess the relationship between different random numbers and Y_i .

5. Conclusions

In this paper, we proposed a secure key agreement scheme for low-energy wireless sensor networks. We made use of negotiatory keys instead of the secret keys that used in almost all the schemes developed so far, and different random numbers instead of a random number. The negotiatory keys prevent the entities of key agreement from revealing unshared secret keys and the number of shared keys. Also, the proposed scheme resolves the weakness of the cryptographic algorithm by exploiting multiple random numbers during a key agreement phase. Consequently, the proposed scheme guarantees that two nodes agree a session key in a secure method and provides the robustness against the compromise of nodes. Simulation results have proven that it does not reduce the key sharing probability between any two nodes providing the high robustness against a node compromise. Also, security analyses indicate that it is more secure than the simple MRS scheme. That is, judging from the security and availability of the proposed scheme, our protocol is extremely suitable for WSNs. In our future work, we will study the communication and computational overhead caused by the proposed scheme and devise an improved scheme for reducing the overheads. Also, the implementation of these techniques on real sensor platforms will be another future research item.

References

1. Cam, H., Ozdemir, S., Muthuavinashiappan, D., Nair, P.: Energy-Efficient Security Protocol for Wireless Sensor Networks. Proc. of IEEE VTC Fall 2003Conference, Oct. 4-9, Orlando, (2003) 2981-2984
2. Eschenauer, L., Gligor, V.D.: A Key-management Scheme for DistributedSensor Networks. Proc. of the 9th ACM Conference on Computer and Communications Security, (2002) 41-47
3. Du, W., Deng, J., Han, Y.S., Varshney, P.K.: A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. Proc. of ACM Conference on Computer and Communications Security (CCS'03) (2003) 42-51
4. Chan, H., Perrig, A., Song, D.: Random Key Predistribution Schemes for Sensor Networks. IEEE Symposium on Research in Security and Privacy, May 11-14 (2003) 197-213
5. Du, W., Deng, J., Han, Y.S., Varshney, P.K.: A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge. IEEE INFOCOM (2004) 586-597
6. Liu, D., Ning, P., Du, W.: Group-Based Key Pre-distribution in Wireless Sensor Networks. Proc. of 10th ACM Conference on Computer and Communications Security (CCS'03), (2003) 11-20
7. Chan, A.C-F.: Distributed Symmetric Key Management for Mobile Ad hoc Networks. IEEE INFOCOM (2004) 2414-2424
8. Rivest, R.L., Adleman, L., Dertouzos, M.L.: On Data Banks and Privacy. In Foundations of Secure Computation, eds. R. A. DeMillo et al., Academic Press, (1978) 169-179.