Computing and Informatics, Vol. 27, 2008, 743–756

GROUP-BASED KEY MANAGEMENT PROTOCOL FOR ENERGY EFFICIENCY IN LONG-LIVED AND LARGE-SCALE DISTRIBUTED SENSOR NETWORKS

Kwang-Jin Paek, Jongwan Kim
Chong-Sun Hwang, SangKeun Lee

Department of Computer Science and Engineering
Korea University
Seoul, Korea
e-mail: {pkj, wany, hwang}@disys.korea.ac.kr, yalphy@korea.ac.kr

Ui-Sung Song

Deptartment of Computer Education
Busan National University of Education
Busan, Korea
e-mail: ussong@bnue.ac.kr

Revised manuscript received 10 January 2008

Abstract. As wireless sensor networks grow, so does the need for effective security mechanisms. We propose a cryptographic key-management protocol, called energy-efficient key-management (EEKM) protocol. Using a location-based group key scheme, the protocol supports the revocation of compromised nodes and energy-efficient rekeying. The design is motivated by the observation that unicast-based rekeying does not meet the security requirements of periodic rekeying in long-lived wireless sensor networks. EEKM supports broadcast-based rekeying for low-energy key management and high resilience. In addition, to match the increasing complexity of encryption keys, the protocol uses a dynamic composition key scheme. EEKM also provides group-management protocols for secure group communication. We analyzed the energy efficiency and security of EEKM and compared it to other key-management protocols using a network simulator.

Keywords: Energy efficiency, key management, security, location-based, sensor network

1 INTRODUCTION

The architecture and design of sensor networks and hardware have progressed significantly in the past few years [8, 9, 10]. Sensor nodes (SNs) are small and have wireless communication capability within short distances. An SN typically contains a wireless transmitter/receiver, and power, sensing, processing, and storage units. A wireless sensor network is comprised of a large number of SNs with limited power, computation, storage, and communication capability.

Here we propose an energy-efficient key-management protocol (EEKM) for large-scale WSNs (Wireless Sensor Networks) that supports a lightweight rekeying mechanism while providing security properties similar to those of pairwise key-sharing schemes [11]. Existing key-management protocols focus mainly on the efficiency of distributing keys and key materials to SNs prior to deployment. EEKM does the same, but also introduces an energy-efficient way to improve scalability, rekeying, and resilience. We investigated a regional group-oriented rekeying strategy and designed merge/split protocols based on this rekeying strategy.

The remainder of this paper is organized as follows. In Section 2, we present an overview of the proposed protocol's architecture and assumptions. In Section 3, we describe the protocol in detail. In Section 4, we evaluate the protocol. The evaluation includes an analysis of the protocol's energy efficiency compared to other key-management protocols and a simulated prototype implementation of a sensor network test bed. Finally, in Section 5, we present our conclusions and recommendations for future work.

2 SENSOR NETWORK ARCHITECTURE

In this paper, we use the sensor network model proposed by LEAP [1] and assume a static sensor network with immobile SNs. The base station (BS) acts as the key server that is assumed to be a laptop-class device with unlimited power. The sensor network consists of a large number of SNs distributed throughout the area of interest. The BS can broadcast a message to all SNs. Each node belongs to its own virtual group (VG) before being randomly scattered throughout the field of interest (Figure 1). After deployment, the sensor network is divided into four square regional groups. Each SN can determine its location during the bootstrap, using a Global Positioning System (GPS).

We assumed that an adversary could eavesdrop on all traffic, inject packets, and/or replay old messages. If a node were compromised, all of its information would be available to the attacker. However, the BS could not be compromised.

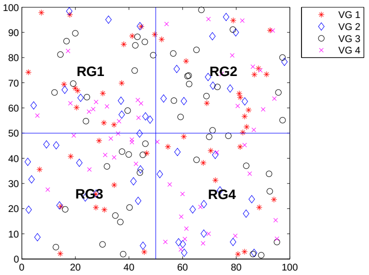

Fig. 1. A 200-node random sensor network with four regional and four virtual groups

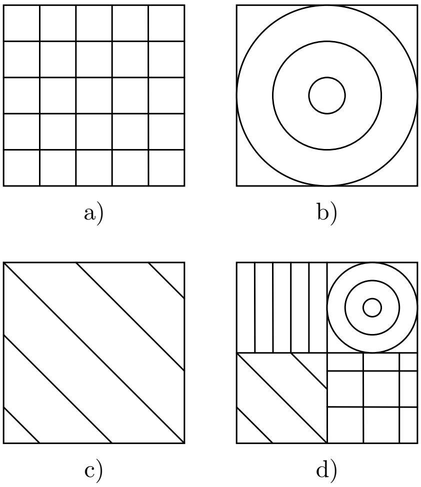

Fig. 2. Patterns of regional groups

Sometimes it is necessary to revoke SNs from a secure network due to node compromise. Therefore, we assumed that there were mechanisms in place to identify compromised SNs [2, 3, 4], and revoke them.

3 ENERGY-EFFICIENT KEY MANAGEMENT PROTOCOL

Table 1 shows the notation used in the EEKM protocol descriptions. To minimize power requirements, we use a MAC pseudo-random function (F) to derive the keys, implemented as $K' = F(K, x) = MAC(K, x)$. SNs are preloaded with an IK, from which further keys can be established.

Notation	Description
BS	Base station of a sensor network
$SEQ(A)$	Message sequence number of A
S_i	Identifier for node i
VG_i	Identifier for virtual group i
RG_i	Identifier for regional group i
RVG_{ij}	Identifier for virtual group j in regional group i
N	Random nonce value
K_{SiBS}	Individual key shared by BS and node i
MK	Master secret key for deriving individual node keys
K_{AB}	Secret key shared by A and B ($K_{AB} = K_{BA}$)
IK	Initial master key for deriving new keys
IKM	Keying material for generating new IK
CK	Common group key shared by nodes and BS
AK	Authentication key for message verification
VK_i	Secret MAC key shared with virtual group i
RK_i	Secret MAC key shared with regional group i
RKM	Keying material for generating new RK
RVK_{ij}	Secret MAC key shared with VG_j in RG_i
$E(K, \ldots)$	Symmetric encryption function using key K
$\|\|$	Concatenation operator
MBK	Material key for deriving KMB
$LOW(KMB[i])$	Low-order half part of $KMB[i]$
$HI(KMB[i])$	High-order half part of $KMB[i]$

Table 1. Notation used in security protocols and cryptographic operations

3.1 Dynamic Key Composition with Key Material Box (KMB)

The *KMB* is generated using the pseudo-random function *F*, and its size can be adjusted to the memory resources of an SN. The larger the size, the more complex the key composition. There is a trade-off between *KMB* memory and the complexity of the dynamic key. However, if the node memory is limited and cannot store the *KMB*, it can compute *KMB* elements on the fly. The computation cost is constant and does not depend on *KMB* size (sz). KMB is as follows: $KMB[i] = F(MBK, i), \{i \mid 1 \leq i \leq sz \text{ and } i \in \mathbb{N}\}$.

3.2 Key Distribution

EEKM key distribution consists of three phases: initialization, group key setup, and pairwise key setup. We use a temporary-master-key approach [1] to generate group and pairwise keys.

3.2.1 Initialization Phase

We use a secret-key mechanism, and each SN stores six keys (K_{SiBS}, IK, AK', CK, VK, and MBK) in the initialization phase. Every node has an individual key that is only shared with the BS. This key is generated and preloaded into each node prior to its deployment. The individual key K_{SiBS} for node S_i (each node has a unique identification) is generated as follows: $K_{SiBS} = F(MK, S_i)$, where F is the pseudo-random function and MK is a master key known only to the BS. When it needs to communicate with an individual node S_i, it computes K_{SiBS} on the fly.

We refer to $F(AK, 0)$ as the verification key AK', which is stored in each node. The equation $AK' = F(AK, 0)$ enables a node to verify the authenticity of a message with AK, a random number. The network-wise key (CK) is used to secure the broadcast messages to all of the SNs. The BS generates an IK and then loads it into each node. CK is generated as follows: $CK = F(IK, 0)$. This is a network-wise key used for broadcasting messages to the entire network.

The virtual group key VK is for randomly classified nodes. Figure 1 shows the randomly distributed nodes of each VG. Each node is classified into an equal number of *VGs*, and has its own virtual group identifier (VG_i). VK is generated as follows: $VK_i = F(IK, VG_i)$. MBK is generated with $F(IK, 1)$ and is used to create the *KMB* (see Section 3.1). The dynamic composition key (DCK) contains the elements of the *KMB* selected by the key composition function (KCF), which uses unique message identification (UMI) as a parameter. UMIs are unique in the lifetime of the WSN and consist of three components: the UID (Unique ID: BS or S_i), message sequence number (MSN; SEQ[BS] or SEQ[S_i]) and material key. The KCF makes up the dynamic secret key with the UMI. The prototype of RGF is *Key KCF(UID, MSN, K).*

The group identification (GID) can be a common group VG_i or RG_i. Node A sends group G an encrypted message with *KCF* and *GK*, as follows. This phase is preformed before deployment.

$$A \rightarrow G : A||SEQ(A), GID, E(KCF(A, SEQ(A), GK_{GID}), message||N)$$

3.2.2 Group Key-Setup Phase

The key-setup phase, performed after deployment stores eight keys in each SN (K_{SiBS}, IK, AK', CK, RK, VK, RVK, and MBK). The SNs of a group share a common location-based group key. The RG identifier is created by using the ID-Generating Function for the Regional Group (RGF). The prototype of RGF is *Id RGF(IK, location, pattern , size, center).*

The regional group key RK is for regionally classified nodes. Figure 1 shows four regional groups. All nodes are regionally divided into *RGs.* Each node has its regional group identifier RG_i, and RK is generated as follows: $RK_i = F(IK, RG_i)$. These patterns are used to effectively isolate compromised nodes and generate an appropriate rekeying message for uninfected groups.

RVG is generated with VG and RG. Each VG in an RG has a unique subgroup ID in the WSN. RVG_{ij} is the subgroup that belongs to RG_i and VG_j; it is different from RVG_{ji}. The number of RVG is $|RG| * |VG|$, where $|x|$ is the number in group x. In Figure 1, the number of RVG is 16. RVG, which improves resilience by dividing nodes into small subgroups, is generated as follows: $RVG_{ij} = F(IK, RG_i||VG_j)$. RVK is created with IK and RVG. It is a subgroup key for RVG and is generated as follows: $RVK_{ij} = F(IK, RVG_{ij})$.

3.2.3 Pairwise Key-Setup Phase

Node A computes its pairwise key with B, K_{AB}, as $K_{AB} = F(K_B, A)$ and $K_B = F(IK, B)$. Node B computes K_{BA} in the same way. K_{AB} serves as their pairwise key after deployment. These steps and neighbor-discovery steps are accomplished simultaneously.

Pairwise key-setup is executed as follows:

$$\begin{aligned} &A \rightarrow \text{ the neighbor nodes of } A(\text{broadcast}) : A \\ &\qquad\qquad\qquad\qquad\qquad B \rightarrow A : B, MAC(K_{AB}, A||B). \end{aligned}$$

When two neighbor nodes, A and B, are added at the same time, the above scheme can be simplified. If A receives B's response to its message before responding to B's message, A will omit its own response. They will have two different pairwise keys, K_{AB} and K_{BA}. If $A < B$, they can choose K_{AB} as their pairwise key. All nodes erase IK at the end of the pairwise key-setup phase.

3.3 Addition and Deletion of Nodes

Before deployment, the new nodes complete an initialization phase and have N^1 and $N^{2\prime} = F(N^2, 0)$ in the pairwise key-setup phase. N^1 and N^2 are nonce used for mutual authentication during pairwise key setup. After deployment, they perform a group key setup and another pairwise key setup.

The pairwise key-setup steps for new nodes are executed as follows:

$$\begin{aligned} BS \rightarrow \text{ the existing nodes } &: BS||SEQ(BS), \\ &\quad E(KCF(BS, SEQ(BS), CK), F(N^1, 0)||N^2||N) \\ \text{new node } A \rightarrow \text{ neighbor nodes } &: A, N^1. \end{aligned}$$

If neighbor nodes are the existing nodes, the pairwise key-setup phase is written as follows:

$$\begin{aligned} &\text{existing neighbor node } B \rightarrow \text{ the new node } A : B, N^2 \\ &\text{new node } A \rightarrow \text{ the existing neighbor node } B : A||SEQ(A), \\ &\qquad\qquad\qquad\qquad\qquad\qquad E(KCF(A, SEQ(A), K_B), K_A||N). \end{aligned}$$

If neighbor nodes are new nodes, the pairwise key-setup phase is identical to the initial pairwise key-setup phase:

$$\text{The existing neighbor node } B \rightarrow \text{ the new node } A : B, MAC(K_{AB}, A||B).$$

After this step, the *IK* of new nodes is erased and pairwise keys are established in all nodes.

Key revocation refers to the task of securely removing keys which are known to be compromised. Existing key revocation schemes can be divided into two categories: centralized key revocation scheme [12, 1] and distributed key revocation scheme [13, 14]. In a centralized key revocation scheme, a centralized authority (BS) is used to revoke compromised sensors [12, 1]. In a distributed key revocation scheme, no centralized authority is used and a vote is cast and collected among sensor nodes. If the vote tally against a sensor node exceeds a specified threshold, the sensor node will be revoked [13, 14]. EEKM belongs to the centralized key revocation scheme. This paper focuses on the centralized key revocation scheme. We compare the centralized revocation schemes proposed in [13], LEAP, and EEKM in Section 4.

It is important to securely update group keys when a compromised node is detected. The group keys must be changed and distributed to all the remaining nodes in a secure, reliable, and timely fashion. This is referred to as group rekeying. The BS broadcasts the revocation message to all nodes. CNODE stands for a compromised node and $\{CNODE_1||CNODE_2||\ldots\}$ is the set of compromised nodes.

$$\begin{aligned} BS \rightarrow \text{ all nodes (broadcast)} : & BS||SEQ(BS), AK^i, F(AK^{i+1}, 0), \\ & E(KCF(BS, SEQ(BS), CK), \\ & \{CNODE_1||CNODE_2||\ldots\}||N) \end{aligned}$$

All nodes authenticate the revocation message with AK^i and $AK^{\prime i} = F(AK^i, 0)$. This message includes the verification key $AK^{\prime i+1} = F(AK^{i+1}, 0)$ for authentication of the next message. All nodes verify the authenticity of the revocation message and then eliminate compromised nodes from the neighbor node list of each SN.

$$\begin{aligned} BS \rightarrow \text{ all nodes (broadcast)} : & BS||SEQ(BS), AK^{i+1}, F(AK^{i+2}, 0), \\ & \{RG_a, E(KCF(BS, SEQ(BS), RK_a), IK^{i+1}||N)|| \\ & RG_b, E(KCF(BS, SEQ(BS), RK_b), IK^{i+1}||N)||\ldots\} \end{aligned}$$

The above group rekey message is used to update the *IK* of all regional groups except for compromised regional groups. All nodes authenticate this message with AK^{i+1} and save $N^{i+2\prime} = F(AK^{i+2}, 0)$ for the next authentication.

If some nodes in RVG_{cc} are compromised, non-compromised $RGVs$ except for RVG_{cc} receive the new IK. The IK update message for the compromised region RG_c is as follows:

$$\begin{aligned} BS \rightarrow \text{ all nodes (broadcast)} : & BS||SEQ(BS), AK^{i+2}, F(AK^{i+3}, 0), \\ & \{RVG_{ca}, E(KCF(BS, SEQ(BS), RVG_{ca}), IK^{i+1}|| \\ & N)||RVG_{cb}, E(KCF(BS, SEQ(BS), RVG_{cb}), \\ & IK^{i+1}||N)|| \ldots\}. \end{aligned}$$

All *RVGs* in RG_c receive a new *IK*, except for RVG_{cc}. The other nodes only save $F(AK^{i+3}, 0)$. *C*, the non-compromised neighbor node of *D*, does not belong to RVG_{cc} and has the new IK^{i+1}. The node *D* can get the new IK^{i+1} from the node *C* as follows:

$$\begin{aligned} & D \rightarrow C : D, MAC(K_{CD}, C||D||N) \\ & C \rightarrow D : C, SEQ(C), E(KCF(C, SEQ(C), K_{CD}), IK^{i+1}||N). \end{aligned}$$

3.4 Key Update

In short-lived networks, the threat can be ignored [5]. For other networks, however, it is necessary to renew the encryption keys occasionally [6].

The rekeying protocol updates the *IK*, and all nodes regenerate each derived key, except for the secret key K_{SiBS} shared between the BS and each sensor node. The following message is broadcast to send a new *IK*.

$$\begin{aligned} BS \rightarrow \text{ all nodes (broadcast)} : & BS||SEQ(BS), AK^i, F(AK^{i+1}, 0), \\ & E(KCF(BS, SEQ(BS), CK), IKM||N) \end{aligned}$$

The new IK^{i+1} is generated by $KCF(SEQ(BS), IK^i, IKM)$. After this broadcast, every derived key generated by IK^i is regenerated with the new IK^{i+1}. To maintain the modified organization of the groups, *RG* and *RVG* are not modified. *RK* and *RVK* do not use the previous equations, but the following equations:

$$RK^{i+1} = F(IK^{i+1}, RK^i), RVK_{ij}^{i+1} = F(IK^{i+1}, RVK_{ij}^i).$$

The regional group key update is carried out as follows:

$$\begin{aligned} BS \rightarrow \textit{all nodes (broadcast)} : & BS||SEQ(BS), AK^i, F(AK^{i+1}, 0), \\ & \{RG_a, E(KCF(BS, SEQ(BS), RG_a), RKM||N)|| \\ & RG_b, E(KCF(BS, SEQ(BS), RG_b), RKM||N)|| \ldots\}. \end{aligned}$$

The new RK^{i+1} is generated by $KCF(SEQ(BS), RK^i, RKM)$. The node belonging to RG_a or RG_b updates its RK^i with the new RK^{i+1}. The new RVK^{i+1} is generated by $F(RKM, RVK^i)$. This message updates *RK* only and does not affect the other keys.

3.5 Group Management: Merging & Splitting

The merge message is sent to groups to integrate them into one group. This message leads to effective group communication.

$$\begin{aligned} BS \rightarrow \text{ all nodes (broadcast)}: & BS||SEQ(BS), AK^i, F(AK^{i+1}, 0), \\ & \{RG_a, E(KCF(BS, SEQ(BS), RG_a), RKM||N) \\ & ||RG_b, E(KCF(BS, SEQ(BS), RG_b), RKM||N) \\ & || \ldots\} \end{aligned}$$

RG^{i+1} is generated by $RGF(RKM, 0, 0, 0, 0)$, and RK^{i+1} is generated by $F(RKM, RG^{i+1})$. Nodes belonging to the target group have the same RK^{i+1}. RVG and RVK do not use the previous equations, but the following equations:

$$RVG_{ij}^{i+1} = F(RKM, RG_i^{i+1}||VG_j),\ RVK_{ij}^{i+1} = F(RKM, RVG_{ij}^{i+1}).$$

The split message is sent to the groups for dividing into proper groups. This message is useful for restricting the effect of a compromised node on the immediate network neighborhood.

$$\begin{aligned} BS \rightarrow \text{ all nodes (broadcast)}: & BS||SEQ(BS), AK^i, F(AK^{i+1}, 0), \\ & \{RG_a, E(KCF(BS, SEQ(BS), RG_a), \\ & RKM||pattern||size||cp||N)|| \ldots\} \end{aligned}$$

The new RG^{i+1} is generated by the $RGF(RKM, ldata, pattern, size, cp)$, the new RK^{i+1} is generated by $F(RKM, RG^{i+1})$, and RVG and RVK are generated by the same equations of the merge message. Each node computes its own RG^{i+1} according to RGF and the parameters. There are various patterns: grid, circle, diagonal, etc. These patterns are applicable to each regional groups. Figure 2 illustrates the various patterns of regional groups.

4 EVALUATION

We simulated EEKM using a network simulator with the random network shown in Figure 1.

We also assume a simple model where the radio dissipates $P_{com} = 50\,\text{nJ/bit}$ to run the transmitter or receiver circuitry and $P_{amp} = 100\,\text{pJ/bit/m}^2$ for the transmit amplifier. We assume the overall distance for transmission to be r, the minimum receiving power at a node for a given transmission error rate is $P_{receive}$, and the power at a transmission node is P_{send}. The radio frequency (RF) attenuation model near the ground is given by $P_{receive} \propto \frac{P_{send}}{r^\alpha}$ where r is the transmission distance and α is the RF attenuation exponent. Due to multiple paths and other interference effects,

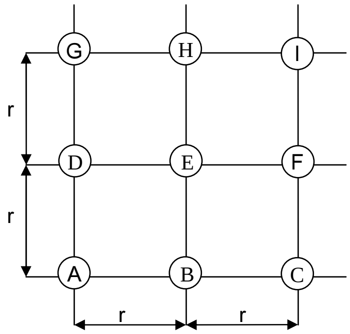

Fig. 3. The normalized sensor network for measuring the energy cost of each protocol

α typically ranges from 2 to 5 [7]. We assume α to be 2. Thus, to transmit a k-bit message with distance r, we use two equations: $P_{send}(k,r) = P_{com} \times k + P_{amp} \times k \times r^2$, $P_{receive}(k) = P_{com} \times k$. Using these equations and the random 200-node network shown in Figure 1, we simulated the transmission of data between every node and sink node that was located within 50 m (at $x = 50$, $y = -50$). For our experiments, we assume that each node receives an 8 192-bit control packet from the sink node for rekeying.

Protocols	Rekeying object	Communication cost	
		Send	Receive
EEKM	Group and pairwise key	0	$9 \times P_{receive}(k)$
LEAP	Group key	$8 \times P_{send}(k,r)$	$9 \times P_{receive}(k)$
Random-key	Pairwise key	$12 \times 3 \times P_{send}(k,r)$	$12 \times 2 \times P_{receive}(k)$

Table 2. Communication cost for rekeying

We analyzed the communication cost of EEKM compared to LEAP and a random graph-based scheme. Table 2 shows the communication costs of rekeying for the three protocols in the normalized sensor network (see Figure 3). In EEKM, the BS broadcasts the newly encrypted *IK* to all nodes. In LEAP, the BS initiates the process by sending the new group key to each of its children in the spanning tree using its cluster key for encryption. In the random-key preconfiguration scheme, rekeying is equivalent to self-revocation of a key by a node. After removing the expired key, the affected nodes restart the discovery process of shared keys, and possibly the path-key establishment phase. Figure 4 plots the remaining energy by using the equations (Table 2) with $r = 1$ and $k = 8192$.

Figure 5 shows the average remaining energy of 10 simulation results with each rekeying protocol. In EEKM, the plot does not change in each simulation, whereas they change using the other protocols. There is difference between two figures, because the cost of EEKM rekeying is topology independent.

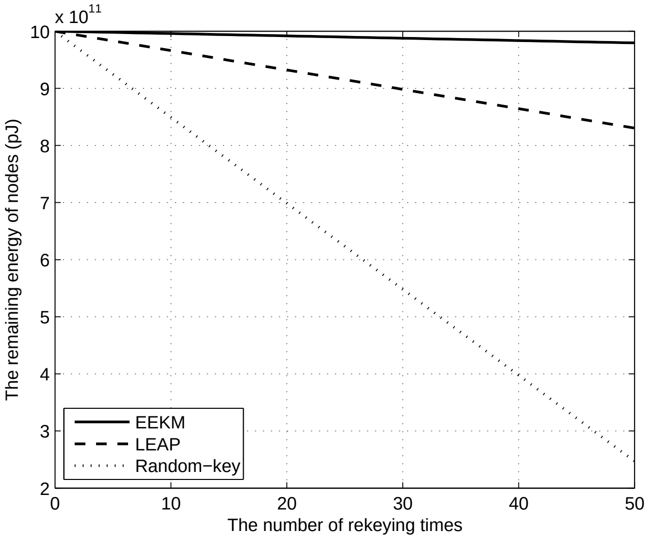

Fig. 4. The remaining energy after sending each rekeying message using the equations in Table 2

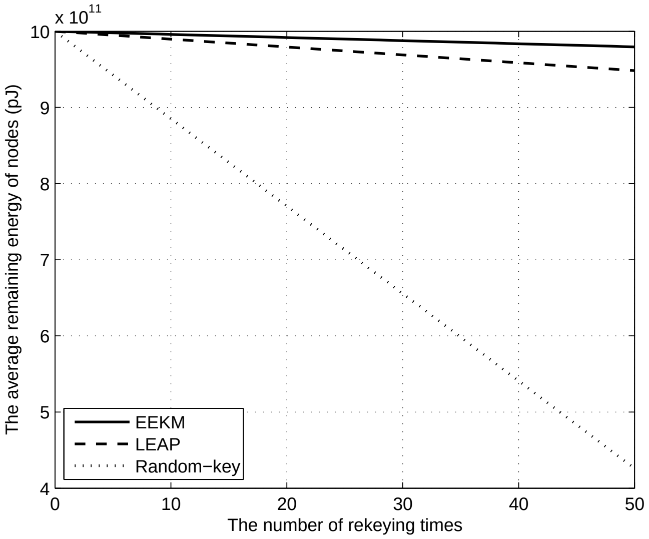

Fig. 5. The average remaining energy of the simulation results after sending each rekeying message

5 CONCLUSIONS AND FUTURE WORK

We designed an energy-efficient key-management (EEKM) protocol for large-scale distributed sensor networks. EEKM uses a predeployed temporary master key approach that supports a robust and lightweight method for setting up various derived keys. A broadcast-based rekeying protocol is suitable for periodic rekeying and long-lived next-generation WSNs. Our simulation results indicate that EEKM is more energy-efficient than the other key-management protocols. EEKM provides group-management protocols for secure group communication. Next-generation sensor networks will be long-lived, highly dynamic, and quality of service (QoS) supportable. The attack profile on these networks will be more varied and complex. Our research is needed on adaptive key management to solve these challenges.

REFERENCES

[1] Zhu, S.—Jajodia, S.: LEAP: Efficient Security Mechanisms for Large-Scale Distributedsensor Networks. Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03), 2003, pp. 62–72.

[2] Buchegger, S.—Boudec, J. L.: Performance Analysis of the CONFIDANT Protocol (Cooperation of Nodes: Fairness in Dynamic Ad-Hoc Networks). Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2002, pp. 226–236.

[3] Marti, S.—Baker, M.: Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking, 2000, pp. 255–265.

[4] Zhang, Y.—Lee, W.: Intrusion Detection in Wireless Ad Hoc Networks. Proceedings of the 6th International Conference on Mobile Computing and Networking (MobiCom 2000), pp. 275–283.

[5] Perrig, A.—Tygar, J. D.: SPINS: Security Protocols for Sensor Networks. Wireless Network, Vol. 8, 2002, No. 5, pp. 521–534.

[6] Fumy, W.—Landrock, P.: Principles of Key Management. IEEE Journal of Selected Areas in Communications, Vol. 11, 1993, pp. 785–793.

[7] Zhao, F.—Guibas, L. J.: Wireless Sensor Networks: An Information Processing Approach. Elsevier, Oxford, UK, 2004.

[8] Akkaya, K.—Younis, M.: A Survey on Routing Protocols for Wireless Sensor Networks. Ad Hoc Networks, Vol. 3, 2005, No. 3, pp. 325–349.

[9] Raghunathan, V.—Srivastava, M. B.: Energy-Aware Wireless Microsensor Network. IEEE Signal Processing Magazine, March 2002, pp. 40–50.

[10] Shih, E.—Chandrakasan, A.: Energy-Efficient Link Layer for Wireless Microsensor Network. Proceedings of the Workshop on VLSI 2001 (WVLS '01), Orlando, Florida, 2001, pp. 16–21.

[11] Paek, K. J.—Song, U. S.: An Energy-Efficient Key Management Protocol for Large-Scale Wireless Sensor Networks. Proceedings of 2007 International Conference on Multimedia and Ubiquitous Engineering (MUE 2007), 2007, pp. 201–206.

[12] Eschenauer, L.—Gligor, V.: A Key-Management Scheme for Distributed Sensor Networks. Proceedings of ACM CCS 2002, 2002, pp. 41–47.

[13] Chan, H.—Song, D.: Random Key Predistribution Schemes for Sensor Networks. Proceedings of the IEEE Security and Privacy Symposium, 2003, pp. 197–213.

[14] Chan, H.—Gligor, V.: On the Distribution and Revocation of Cryptographic Keys in Sensor Networks. IEEE Transactions on Dependable and Secure Computing, Vol. 2, No. 3, July-September 2005, pp. 197–213.

Kwang-Jin Paek received the B. Sc. and M. Sc. degrees in electronic and computer engineering from Pusan University of Foreign Studies, South Korea, in 1996 and 1998, respectively, and the Ph. D. degree in computer science and engineering from Korea University, South Korea, in 2007. He is currently a postdoctoral in Embedded Software Platform Research Group, ETRI, South Korea. His research interests include WSN, MAC protocols in WSN, key management protocols in WSN, middleware systems, and mobile agent systems.

Jongwan Kim received the Ph. D. degree in computer science and engineering from Korea University, South Korea, in 2007, B. Sc. degree in business administration, and M. Sc. in computer science and engineering from Shamyook University, Soongsil University. He has more than 10 years field experiences as a developer and technician in object-oriented technology. His research interests include mobile & streaming data management, location-based services, sensor/RFID, and object oriented technologies.

Chong-Sun Hwang received the M. Sc. degree in mathematics from Korea University, South Korea, in 1970, and the Ph. D. degree in statistics and computer science from the University of Georgia in 1978. From 1978 to 1980, he was an Associate Professor at South Carolina Lander State University. He is currently a full professor in the College of Information and Communication at Korea University, South Korea. His research interests include distributed systems, distributed algorithms, and mobile computing systems.

SangKeun Lee is a corresponding author. He received the B. Sc., M. Sc., and Ph. D. degrees in computer science and engineering from Korea University, South Korea, in 1994, 1996, and 1999, respectively. Since 2003, he has been an Assistant/Associate Professor in the College of Information and Communication, Korea University, South Korea. His research interests include data management in mobile/pervasive computing systems, location-based information systems, XML databases, and data management in mobile ad hoc networks.

Ui-Sung Song received his M. Sc. and Ph. D. degrees in computer science and engineering from Korea University, Seoul, Korea in 1999 and 2005, respectively. He is currently an instructor in the Department of Computer Education at the Busan National University of Education. His recent research interests include distributed and mobile computing, sensor network and network security.