Secure Group Communication with Self-healing and Rekeying in Wireless Sensor Networks

Firdous Kausar[1], Sajid Hussain[2], Jong Hyuk Park[3], and Ashraf Masood[1]

[1] College of Signals, National University of Science and Technology (NUST), Rawalpindi, Pakistan
firdous.imam@gmail.com, ashrafm61@gmail.com
[2] Jodrey School of Computer Science, Acadia University, Nova Scotia, Canada
sajid.hussain@acadiau.ca
[3] Department of Computer Engineering, Kyungnam University, Masan, Korea
parkjonghyuk@gmail.com

Abstract. We have developed a self-healing key distribution scheme for secure multicast group communications for wireless sensor network environment. We present a strategy for securely distributing rekeying messages and specify techniques for joining and leaving a group. Access control in multicast system is usually achieved by encrypting the content using an encryption key, known as the group key (session key) that is only known by the group controller and all legitimate group members. In our scheme, all rekeying messages, except for unicast of an individual key, are transmitted without any encryption using one-way hash function and XOR operation. In our proposed scheme, nodes are capable of recovering lost session keys on their own, without requesting additional transmission from the group controller. The proposed scheme provides both backward and forward secrecy. We analyze the proposed scheme to verify that it satisfies the security and performance requirements for secure group communication.

Keywords: sensor networks, security, key distribution, secure group communication, one-way hash chains.

1 Introduction

Wireless sensor network (WSN) consists of a large number of small, low cost sensor nodes which have limited computing and energy resources. Usually, sensor nodes perform in-network processing by reducing large streams of raw data into useful aggregated information. Therefore, compromised nodes could deviate the network behavior by injecting false data or modifying data of correct nodes. Thus, it must be guaranteed that compromised nodes do not take part in the group communication.

Secure group communication needs a secret shared by all the group members for group oriented applications in wireless sensor networks (WSNs). The shared key provides group secrecy and source authentication. A single symmetric key known only to the group members can effectively protect a multicast group.

H. Zhang et al. (Eds.): MSN 2007, LNCS 4864, pp. 737–748, 2007.

However, only legitimate users should have access to the group communication in order to achieve privacy [1]. In rekeying, the session keys are updated periodically, when new users join or old users leave the group. The keys are securely redistributed to the existing members of the group in order to provide forward secrecy (FS) as well as backward secrecy (BS). The newly joined users should not be able to derive the previous group keys, even if they are able to derive future group keys with subsequently distributed keying information. Similarly, the revoked users should not be able to derive the future session keys, even if they are able to compute the previous session keys with previously distributed keying information.

The rekeying is performed by the group controller (GC). The most important parameters when performing group rekeying are as follows: the number of keys stored by the group controller, the number of keys stored by each group member, the number of keys delivered in the initialization stage, bandwidth required for updating the keys, and latency for updating the session key [2].

As the size of the group grows and/or the rate of membership change increases, the frequency of rekeying becomes the primary bottleneck for rekeying on each membership change. Therefore, scalable group rekeying is an important and challenging problem to be addressed in order to support secure multicast communication.

Another important problem in multicast communication is reliability. Since multicasting is an unreliable mode of communication, packets may be lost during the communication. If a packet containing key updating information is lost, authorized receivers may not be able to calculate the session key. This may influence rekeying and so the rekeying system must be self-healing if packet loss occurs. In a large and dynamic group communication over an unreliable network, the main concept of self-healing in key distribution schemes is that users can recover lost session keys on their own, without requesting additional transmissions from the group manager, even if some previous key distribution messages are lost. This reduces network traffic, the risk of user exposure through traffic analysis, and the work load on the group manager.

The key idea of self-healing key distribution schemes is to broadcast information that is useful only for trusted members. Combined with its pre-distributed secrets, this broadcast information enables a trusted member to reconstruct a shared key. On the contrary, a revoked member is unable to infer useful information from the broadcast. The only requirement that a user must satisfy to recover the lost keys through self-healing is its membership in the group both before and after the sessions in which the broadcast packet containing the key is sent. A user who has been off-line for some period is able to recover the lost session keys immediately after coming back on-line. Thus self-healing approach of key distribution is stateless.

The need to form a group might be driven by the query being propagated through a node. As a result, a node may need to define a multicast group to make the query initiated in those nodes and then collect the result efficiently

and securely. Further, a node may also modify such queries effectively over the time. For instance, a multicast group could be a region defined with a geometric shape.

This paper provides a computationally secure and efficient group key distribution scheme with self-healing property and time-limited node revocation capability for large and dynamic groups over insecure WSNs. The session keys are updated periodically, where the update is performed regardless of changes in network (group) topology. Periodic rekeying can significantly reduce both the computation and communication overhead at the GC and the nodes, and thus improve the scalability and performance of key distribution protocols. It is shown that the proposed scheme can tolerate high channel loss rate, and hence make a good balance between performance and security, which is suitable for WSN applications.

The paper is organized as follows. In Section 2, related research is described. In Section 3, we describe the preliminaries assumed throughout the paper. We describe the security properties in Section 4, and give details of our proposed scheme in Section 5. In Section 6, we present an analysis of proposed scheme. Finally, we summarize our paper in Section 7.

2 Related Work

Recently there have been several proposals to address the secure group communication issues. The most known technique is the construction of a logical key tree where group members are associated with leaves and each member is given all the keys from his leaves to the root, as proposed in [3] [4] [5] [6], where root key is the group key. This approach allows reducing the communication cost for key update, on the event of group membership change, to $O(logM)$ where M is the number of group members.

Several extensions are proposed to deal with reliability [7], node dependent group dynamics [8], and time variant group dynamics [9]. Extensions to wireless networks are discussed in [10] and several secure multicast protocols are proposed in [11] [12].

Park et al. [13] propose a lightweight security protocol(LiSP) for efficient rekeying in dynamic groups. LiSP utilizes broadcast transmission to distribute the group keys and uses one-way key chains to recover from lost keys. While this scheme is very efficient, LiSP requires the use of static administration keys to perform periodic administrative functions. This leaves those keys vulnerable to disclosure.

Wong et al. [14] propose the the group re-keying, which relies only on current rekeying message and the node's initial configuration. A non-revoked node can decrypt the new session keys independently from the previous re-keying messages without contacting the GC, even if the node is off-line for a while. They use keys of multiple granularity to reduce the rekeying overhead associated with membership management.

Carman et al. [15] give a comprehensive analysis of various group key schemes and find that the group size is the primary factor that should be considered when choosing a scheme for generating and distributing group keys in a WSN.

Staddon et al. [16] propose a self-healing group key distribution scheme based on two-dimension t-degree polynomials. Liu et al. [17] further improve the work in [16] by reducing the broadcast message size in situations where there are frequent but short-term disruptions of communication, as well as long-term but infrequent disruptions of communication. Blundo et al. [18] also present a design of self-healing key distribution schemes which enables a user to recover from a single broadcast message where all keys are associated with sessions where it is a member of the communication group.

Jiang et al. [19] propose a key distribution scheme with time-limited node revocation based on dual directional hash chains for WSNs. Dutta et al. [20] propose two constructions for self-healing key distribution based on one-way hash key chains with t revocation capability using polynomial based node revocation.

3 Preliminaries

Definition 1. *A one-way hash function H can map an input M of the arbitrary length to an output of the fixed length, which is called hash value $h : h = H(M)$, where the length of M is m-bits. One-way hash function H has the following properties [21]:*

- *Given a hash value h, it is computationally infeasible to find the input M such that $H(M) = h$*
- *Given an input M, it is computationally infeasible to find a second input $\acute{M}$ such that $H(\acute{M}) = h$, where $\acute{M} \neq M$*

Definition 2. *Let H be a one-way hash function and s be a random seed. Then, a hash chain can be deduced by iteratively hashing s, which can be written as: $H^i(s) = H(H^{(i-1)}(s)), i = 1, 2, \ldots$ where, s is regarded as "trust anchor" of the one-way hash chain. The hash chain includes a sequence of hash values, which can be denoted by $h_1 = H(s), h_2 = H(h_1), \ldots, h_i = H(h_{i-1}), i = (1, 2, \ldots)$*

Definition 3. *Let $G(x, y) = H(x) \oplus y$, where H is a one-way hash function and $\oplus$ denotes the bitwise XOR. Given x and $G(x, y)$, without the knowledge of y, it is computationally infeasible to find $\acute{y}$ such that $G(x, \acute{y}) = G(x, y)$*

Node Revocation. The concept of node revocation can be described as follows. Let G be the set of all possible group nodes, and R be the set of revoked nodes, where $R \subseteq G$. The group node revocation is required to offer a secure way for GC to transmit rekeying messages over a broadcast channel shared by all nodes so that any node $n_i \in \{G - R\}$ can decrypt the rekeying messages, whereas any node in R, $n_i \in R$, cannot decrypt rekeying messages.

Session Key Distribution with Confidentiality. The confidentiality in the session key distribution requires that for any node n_i, the session key is efficiently determined from the personal secret of n_i and the broadcasted rekeying message from GC. However, for any node in R it is computationally infeasible to determine the session key. What any node n_i learns from broadcast rekeying message, it cannot be determined from broadcasts or personal keys alone. Let a group of k nodes is defined as $n_1, n_2, \ldots, n_k$. If we consider separately either the set of m broadcasts $\{B_1, \ldots, B_m\}$ or the set of k personal keys $\{S_1, \ldots, S_k\}$, it is computationally infeasible to compute session key SK_j (or other useful information) from either set.

Let m denote the total number of sessions in the life cycle of the group communication. Each node is assigned a pre-arranged life cycle (t_1, t_2), which depends on the time of joining. In other words, it can be said that each node will participate in the group communication for $k = t_2 - t_1$ number of sessions. Due to which, once a node's life cycle is expired, it is automatically detached from the group session without requiring the direct intervention of the GC.

For a group with life cycle $(0, m)$, the group key for session j is as follows:

$$SK_j = K_j^F + K_{m-j+1}^B \tag{1}$$

where K_j^F is the forward key and K_{m-j+1}^B is the backward key for session j.

4 Security Properties

A rekeying scheme should provide the following types of security.

Definition 4. *A rekeying protocol provides forward secrecy if for any set $R \subseteq G$, where all $n_l \in R$ are revoked before session j, it is computationally infeasible for the members in R to get any information about SK_i for all $i \geq j$, even with the knowledge of session keys $\{SK_1, \ldots, SK_{j-1}\}$ before session j.*

Definition 5. *A rekeying protocol provides backward secrecy if for any set $J \subseteq G$, where all $n_l \in J$ are newly joined nodes after session j, it is computationally infeasible for the members in J to get any information about SK_i for all $i \leq j$, even with the knowledge of group keys $\{SK_{j+1}, \ldots, SK_m\}$ after session j.*

Definition 6. *A rekeying protocol is key-independent, if it is both forward-secret and backward-secret.*

5 Our Proposed Scheme

In this section, we provide the details of our proposed scheme of self-healing key distribution with time limited node revocation capability. First, nodes are divided into groups, where each group is managed by the group controller (GC). However, the groups are dynamic and regrouping is done after specific duration. The details of group formation is not discussed in this paper. The group life

cycle is given by m, which determines the total number of sessions for a group. The GC uses the pseudorandom number generator (PRNG) of a large enough period to produce a sequence of m random numbers $(r_1, r_2, \ldots, r_m)$. The GC randomly picks two initial key seeds, the forward key seed S^F and the backward key seed S^B. In the pre-processing time, it computes two hash chains of equal length m by repeatedly applying the same one-way hash function on each seed. For $K_0^F = S^F$ and $K_0^B = S^B$, the hash sequences are generated as follows:

$$\{K_0^F, H(K_0^F), \ldots, H^i(K_0^F), \ldots, H^{m-1}(K_0^F), H^m(K_0^F)\}$$

$$\{K_0^B, H(K_0^B), \ldots, H^i(K_0^B), \ldots, H^{m-1}(K_0^B), H^m(K_0^B)\}$$

During the initial configuration setup, each node n_i is first assigned a prearranged life cyle (t_1, t_2) where $t_1 \geq 1$ and $t_2 \leq m$. n_i will participate in the group communication $k = t_2 - t_1 + 1$ number of sessions. The node n_i joins the group at time t_1 in session p and will have to leave the group at time t_2 in session q, where $q > p$.

The node n_i receives its personal secret from GC consisting of: 1) a forward key in session p i.e. K_p^F, and 2) k number of random numbers corresponding to the sessions in which node n_i will participate in the group communication. Further, GC securely sends the personal secret to n_i using key encryption key KEK_i shared between n_i and GC, as shown below:

$$GC \rightarrow n_i : E_{KEK_i}(K_p^F, (r_p, r_{p+1}, \ldots, r_q)), MAC(K_p^F \| (r_p, r_{p+1}, \ldots, r_q))$$

The node n_i decrypts the message by its corresponding KEK_i to retrieve its secret. In the j-th session the GC locates the backward key K_{m-j+1}^B in the backward key chain and computes the broadcast message

$$B_j = G(K_{m-j}^B, r_j) \tag{2}$$

When the nodes receive the broadcast message B_j, the session key is generated as follows:

– First, when any node n_i in the group receives the broadcast message, it recovers the backward key K_{m-j+1}^B for session j from B_j, by applying XOR on both B_j and r_j, as given below:

$$K_{m-j+1}^B = B_j \oplus r_j \tag{3}$$

From Equation 2 and Equation 3:

$$K_{m-j+1}^B = G(K_{m-j}^B, r_j) \oplus r_j \tag{4}$$

By substituting the value of G i.e. $G(x, y) = H(x) \oplus y$, backward key is given as follows:

$$K_{m-j+1}^B = H(K_{m-j}^B) \oplus r_j \oplus r_j \tag{5}$$

The backward key is obtained:

$$K_{m-j+1}^B = H(K_{m-j}^B) \tag{6}$$

– Second, the node n_i computes the $j-th$ forward key by applying one-way hash function on its forward key K_p^F as follows:

$$K_j^F = H^{j-p}(K_p^F) \tag{7}$$

– Finally, the node n_i computes the current session key SK_j as follows:

$$SK_j = K_j^F + K_{m-j+1}^B \tag{8}$$

5.1 Adding a Group Member

When a node n_i wants to join an active group, the corresponding actions (steps) are described as follows:

– The node n_i obtains the permission to attach to the group communication from the GC. If it is successful, n_i establishes a common secret key KEK_i shared with the GC.
– GC assigns a life cycle to n_i i.e. t_1, t_2.
– Then, the GC sends the current system configuration to n_i using an Init-GroupKey message, as shown below:

$$GC \rightarrow n_i : E_{KEK_i}(K_p^F, (r_p, r_{p+1}, \ldots, r_q)), MAC(K_p^F \| (r_p, r_{p+1}, \ldots, r_q))$$

where K_p^F and $(r_p, r_{p+1}, \ldots, r_q)$ are shared personal secret for n_i with life cycle (t_1, t_2)
– Upon receiving the broadcast message from GC, n_i computes the current session key and participates in the network communication.

5.2 Node Revocation

A node with a life cycle (t_1, t_2) detaches from the group at time t_2. Figure 1 shows a time line to illustrate node life cycle and revocation. The node participates only between t_1 and t_2, where the interval is divided into a l number of sessions. Further, as each node is assigned with a $l = t_2 - t_1 + 1$ number of random numbers, it cannot derive the session keys $SK_t = K_t^F + K_{m-t+1}^B$ for $t < t_1$ and $t > t_2$. These l random numbers correspond to the sessions in which the node participate in the group. So, these random numbers can be used for specified sessions only and cannot be used for the remaining sessions. In order to recover K_{m-t+1}^B at time t from the B_t, it requires r_t, which is not available. Thus, a time limited node revocation is achieved implicitly without any intervention from the GC. As a result, the communication and the computation overhead on the GC and group nodes are remarkably reduced.

Compromised Node. If a compromised node is detected, all nodes are forced to be re-initialized. Let n_i with life cycle (t_1, t_2) is compromised in session k, where $t_1 < k < t_2$, as shown in Figure 1. The GC re-initializes the group communication system by re-computing a new random number sequence of length $t_2 - k + 1$ and then unicast it to all group nodes securely.

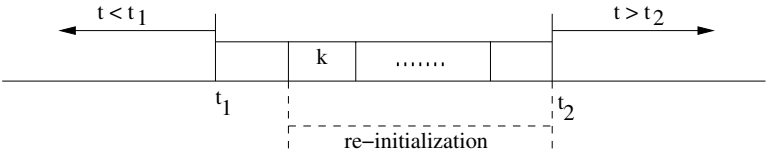

Fig. 1. Node Revocation

6 Analysis

In this section we show that the proposed scheme realizes self-healing key distribution scheme with time limited revocation capability. Further, the forward and backward secrecy is assured with the time-limited node revocation.

We consider a group of sensor nodes as G, $G = \{n_1, n_2, \ldots, n_N\}$, R represents a set of revoked nodes $R \subseteq G$, J represents a set of newly joining nodes $J \subseteq G$, and m is the total number of sessions in the group life cycle.

6.1 Self-healing Property

Consider a node $n_k \in G$ with life cycle (t_1, t_2), which means that n_k joins the group at t_1 (session p) and leaves the group at time t_2 (session q), where $1 \leq p \leq q$, as shown in Figure 2.

Suppose node n_k goes offline in session $p+1$ and comes online again in session $p + j$ where $(p + j) < q$, as shown in Figure 2. As a result, the node n_k will miss the broadcast messages $B_{p+1} \cdots B_{p+j-1}$ from GC; hence, the session keys $SK_{p+1} \cdots SK_{p+j-1}$ will not be available. When node n_k comes online in session $p+j$, it receives the broadcast message B_{p+j} from GC and recovers the backward key $K^B_{m-(p+j)+1}$ for session $p + j$. So, it can obtain the sequence of backward keys $\{K^B_{m-(p+j-1)+1} \ldots K^B_{m-(p+1)+1}\}$ by repeatedly applying H on $K^B_{m-(p+j)+1}$. The node n_k also holds the forward key $K^F_p = H^p(K^F_0)$ of the session p, and hence can obtain the sequence of forward keys $\{K^F_{p+1}, \ldots, K^F_{p+j}\}$ by repeatedly applying H on K^F_p. Now n_k can find all the session keys from session $p + 1$ to session $p + j$ without requiring any extra information from GC.

6.2 Key Independence

The proposed scheme also meets the security requirement for forward and backward secrecy, which gives key independence. Informally, forward-secrecy means that the compromise of one or more secret keys does not compromise previous secret keys. Likewise, backward-secrecy refers to that the compromise of one or more secret keys does not compromise future secret keys. Key-independence means that the secret keys used in different sessions are basically independent. Thus, even if the attacker finds out the secret key of a certain session, it does not give any advantage in finding the secret keys of other sessions.

Session #	Forward key	Backward Key	
1	$H(K_0^F)$	$H^m(K_0^B)$	
2	$H^2(K_0^F)$	$H^{m-1}(K_0^B)$	
$\vdots$	$\vdots$	$\vdots$	
p	$H^p(K_0^F)$	$H^{m-(p)+1}(K_0^B)$	$\longleftarrow t_1$
$p+1$	$H^{p+1}(K_0^F)$	$H^{m-(p+1)+1}(K_0^B)$	offline
$\vdots$	$\vdots$	$\vdots$	
$p+j-1$	$H^{p+j-1}(K_0^F)$	$H^{m-(p+j-1)+1}(K_0^B)$	
$p+j$	$H^{p+j}(K_0^F)$	$H^{m-(p+j)+1}(K_0^B)$	online
$\vdots$	$\vdots$	$\vdots$	
q	$H^q(K_0^F)$	$H^{m-(q)+1}(K_0^B)$	$\longleftarrow t_2$
$\vdots$	$\vdots$	$\vdots$	
$m-1$	$H^{m-1}(K_0^F)$	$H^2(K_0^B)$	
m	$H^m(K_0^F)$	$H(K_0^B)$	

Fig. 2. Self-healing in node life cycle

Forward Secrecy. It is shown that a single revoked node or a collusion of revoked nodes cannot learn anything about the future group keys since the secrets they knew while they were authorized member of the group will no longer be used in any future rekeying message.

Let R be the set of revoked nodes and all nodes $n_k \in R$ are revoked before the current session j. The node n_k cannot get any information about the current session key SK_j even with the knowledge of $\{SK_i, SK_{i+1}, \ldots, SK_{j-1}\}$ before session j, where i is the earliest session of all the nodes in R, or in other words, i is the minimum for all t_1's of nodes in R. In order to find SK_j, node n_k needs the random number r_j of that session and that r_j will not be available to n_k. Also, because of the one-way property of H, it is computationally infeasible to compute $K_{j_1}^B$ from $K_{j_2}^B$ for $j_1 < j_2$. The nodes in R may know the sequence of backward keys $K_m^B, \ldots, K_{m-j+2}^B$; however, they cannot compute K_{m-j+1}^B in order to find current session key SK_j.

Backward Secrecy. Let J is the set of nodes that join the group in session j. The collusion of newly joining nodes cannot get any information about any previous session keys before session j even with the knowledge of group keys after session j. Each $n_k \in J$ when joins the group, GC gives it $j-th$ forward key i.e. K_j^F, instead of initial forward seed K_0^F. As $K_j^F = H(K_{j-1}^F)$, it is computationally infeasible for n_k to compute the previous forward keys which are required to compute session keys before current session j. Hence, the proposed scheme is backward secure.

Table 1. Time and memory requirements for Tmote Sky

Algorithm	Time (seconds)	RAM (bytes)	ROM (bytes)	Energy (Joules)
SHA-1	10.545×10^{-3}	128	4,048	56.94×10^{-6}
MD5	5.757×10^{-3}	176	12,500	31.09×10^{-6}

6.3 Storage Requirements

In our scheme the GC and all nodes do not need any encryption/decryption process of a re-keying message to update session keys. All computation needed for re-keying is one-way hash function and XOR operation, and all information needed for re-keying is in the current transmission and the initial information.

We implement two one-way hash algorithms, SHA-1 and MD5 using nesC [22] programming language in TinyOS for Moteiv's Tmote Sky sensors. We have considered voltage level of 3 volts and nominal current (with Radio off) as 1.8×10^{-3} amps, as given in Tmote Sky's data sheet [23]. We take data stream of 64 bytes. As shown in Table 1, for SHA-1 the code consumes 128 bytes of RAM, 4048 bytes of ROM, takes approximately 10.5 ms to produce a 160-bit hash of a 64-byte message, and the energy consumption is 56.94 μJoules. MD5 produces a 128-bit message digest for a given data stream. The code consumes 176 bytes of RAM, 12.5 KB of ROM, takes approximately 5.75 ms to hash a message of 64 bytes using 64-byte blocks, and the energy consumption is 31.09 μJoules. The above implementation shows that SHA-1 consumes less memory than MD5; however, it's processing overhead is almost double than MD5.

7 Conclusion

Efficient solutions for the problem of key distribution are essential for the feasibility of secure group communication in sensor networks. In this paper, we develop a key distribution scheme for secure group communication in WSNs. The scheme provides a self-healing mechanism for session key-recovery on possible packet loss in the lossy environment using one-way key chain.

Other features include periodic re-keying of group key and time-limited group node revocation. The session keys are updated periodically, where the update is performed regardless of changes in network (group) topology. Periodic rekeying significantly reduces both the computation and communication overhead at the GC and the nodes, and thus improves the scalability and performance of the proposed scheme. Further, the time-limited node revocation is achieved without any intervention from the GC.

The analysis shows that the proposed scheme is computationally secure and meets the security requirements for forward and backward secrecy. The implementation of two one-way hash algorithms SHA-1 and MD5 on resource constraint sensor nodes (Tmote Sky) shows the feasibility of the proposed scheme

for current wireless sensor network technology. Hence, the scheme results scalable, and particularly attractive for large dynamic groups.

Acknowledgment

This work is in part supported by Higher Education Commission (HEC) Pakistans International Research Support Initiative Programs scholarship given to Firdous Kausar to conduct her research at Acadia University, Canada. Further, we would like to thank National Science and Engineering Research Council (NSERC) Canada for their support in providing RTI and Discovery grants to Dr. Hussain at Acadia University, Canada. This research is also supported by Kyungnam University.

References

1. Wallner, D., Harder, E., Agee, R.: Key management for multicast: Issues and architectures (1999)
2. Canetti, R., Garay, J., Itkis, G., Micciancio, D., Naor, M., Pinkas, B.: Multicast Security: A Taxonomy and Some Efficient Constructions. In: INFOCOMM 1999 (1999)
3. Kurnio, H., Safavi-Naini, R., Wang, H.: A secure re-keying scheme with key recovery property. In: Proceedings of the 7th Australian Conference on Information Security and Privacy, pp. 40–55. Springer, London, UK (2002)
4. Wang, L., Wu, C.K.: Authenticated group key agreement for multicast. In: The 5th International Conference on Cryptology and Network Security, Springer, Heidelberg (2006)
5. Ki, J.H., Kim, H.J., Lee, D.H., Park, C.S.: Efficient multicast key management for stateless receivers. In: Lee, P.J., Lim, C.H. (eds.) ICISC 2002. LNCS, vol. 2587, pp. 497–509. Springer, Heidelberg (2003)
6. Pegueroles, J., Bin, W., Soriano, M., Rico-Novella1, F.: Group rekeying algorithm using pseudo-random functions and modular reduction. In: Li, M., Sun, X.-H., Deng, Q.-n., Ni, J. (eds.) GCC 2003. LNCS, vol. 3032, pp. 875–882. Springer, Heidelberg (2004)
7. Yang, Y.R., Li, X.S., Zhang, X.B., Lam, S.S.: Reliable group rekeying: a performance analysis. In: SIGCOMM 2001. Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, pp. 27–38. ACM Press, New York, NY, USA (2001)
8. Poovendran, R., Baras, J.S.: An information theoretic analysis of rooted-tree based secure multicast key distribution schemes. In: Wiener, M.J. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 624–638. Springer, Heidelberg (1999)
9. Noubir, G., Zhu, F., Chan, A.H.: Key management for simultaneous join/leave in secure multicast. In: Proceedings of MILCOM (2003)
10. Gong, L., Shacham, N.: Multicast security and its extension to a mobile environment. Wirel. Netw. 1(3), 281–295 (1995)
11. Bruschi, D., Rosti, E.: Secure multicast in wireless networks of mobile hosts: protocols and issues. Mob. Netw. Appl. 7(6), 503–511 (2002)

12. Kostas, T., Kiwior, D., Rajappan, G., Dalal, M.: Key management for secure multicast group communication in mobile networks. In: Proceedings of DARPA Information Survivability Conference and Exposition (2003)
13. Park, T., Shin, K.G.: Lisp: A lightweight security protocol for wireless sensor networks. Trans. on Embedded Computing Sys. 3(3), 634–660 (2004)
14. Wong, C.K., Gouda, M., Lam, S.S.: Secure group communications using key graphs. IEEE/ACM Trans. Netw. 8(1), 16–30 (2000)
15. Carman, D., Matt, B., Cirincione, G.: Energy-efficient and low-latency key management for msn networks. In: Proceedings of 23rd Army Science Conference, Orlando FL (2002)
16. Staddon, J., Miner, S., Franklin, M., Balfanz, D., Malkin, M., Dean, D.: Self-healing key distribution with revocation. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 241–257 (2002)
17. Liu, D., Ning, P., Sun, K.: Efficient self-healing group key distribution with revocation capability. In: CCS 2003. Proceedings of the 10th ACM conference on Computer and communications security, pp. 231–240. ACM Press, New York, NY, USA (2003)
18. Blundo, C., Darco, P., Santis, A.D., Listo, M.: Design of self-healing key distribution schemes. Des. Codes Cryptography 32(1-3), 15–44 (2004)
19. Jiang, Y., Lin, C., Shi, M., Shen, X.: Self-healing group key distribution with time-limited node revocation for wireless sensor networks. Ad Hoc Networks 5(1), 14–23 (2007)
20. Dutta, R., Chang, E.C., Mukhopadhyay, S.: Efficient self-healing key distribution with revocation for wireless sensor networks using one way key chains. In: ACNS 2007. Proceedings of 5 th International Conference on Applied Cryptography and Network Security (2007)
21. NIST: Secure hash standard. In: National Institute for Standards and Technology, Gaithersburg, MD, USA (April 1995)
22. Gay, D., Levis, P., von Behren, R., Welsh, M., Brewer, E., Culler, D.: The nesc language: A holistic approach to networked embedded systems. SIGPLAN Not. 38(5), 1–11 (2003)
23. `http://www.moteiv.com/products/docs/tmote-skydatasheet.pdf`