12-2009

Computationally Secure Hierarchical Self-Healing Key Distribution for Heterogeneous Wireless Sensor Networks

Yanjiang YANG
Singapore Management University, yjyang@smu.edu.sg

Jianying Zhou
Singapore Management University

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Feng Bao
Singapore Management University, fbao@smu.edu.sg

DOI: https://doi.org/10.1007/978-3-642-11145-7_12

Computationally Secure Hierarchical Self-healing Key Distribution for Heterogeneous Wireless Sensor Networks

Yanjiang Yang[1], Jianying Zhou[1], Robert H. Deng[2], and Feng Bao[1]

[1] Institute for Infocomm Research, Singapore
{yyang,jyzhou,baofeng}@i2r.a-star.edu.sg
[2] School of Information Systems, Singapore Management University
robertdeng@smu.edu.sg

Abstract. Self-healing group key distribution is a primitive aimed to achieve robust key distribution in wireless sensor networks (WSNs) over lossy communication channels. However, all the existing self-healing group key distribution schemes in the literature are designed for homogenous WSNs that do not scale. In contract, heterogeneous WSNs have better scalability and performance. We are thus motivated to study self-healing group key distribution for heterogeneous WSNs. In particular, we propose the concept of *hierarchical* self-healing group key distribution, tailored to the heterogeneous WSN architecture; we further revisit and adapt Dutta *et al.*'s model to the setting of hierarchical self-healing group key distribution, and propose concrete schemes that achieve computational security and high efficiency.

Keywords: Heterogeneous wireless sensor network, self-healing group key distribution, scalability.

1 Introduction

Wireless sensor networks (WSNs) have a wide range of potential applications, such as battlefield surveillance, wildlife tracking, healthcare monitoring, and natural disaster monitoring. A WSN consists of a large number of sensor nodes, each being a small sensing device capable of collecting and reporting environmental data to base station. Sensor nodes are extremely constrained in hardware, having limited computation capability, storage capacity, and radio transmission range. Worse yet, sensor nodes are usually powered by batteries, restricted power supply is thus yet another major limitation of WSNs.

As WSNs are often deployed where there is no network infrastructure support, they are easily susceptible to adversaries who can intercept or interrupt the wireless communications. It is thus crucial to ensure secure communication when a WSN is used for mission-critical applications. A fundamental service to achieve secure communication is key distribution, whereby sensor nodes establish (secret) keys, to be used to encrypt and authenticate messages. Unfortunately,

S. Qing, C.J. Mitchell, and G. Wang (Eds.): ICICS 2009, LNCS 5927, pp. 135–149, 2009.

it is commonly acknowledged that key distribution in WSNs is not trivial, considering the resource-constrained nature of sensor nodes. Hence lots of efforts have been dedicated to the study of key management and distribution in WSNs [6,7,9,8,12,15,18,19,20,21,27,28,31]. These methods are categorized into group key distribution [6,9,15,20,21] and pairwise key distribution [7,8,12,18,19,27,31]. The former enables a group of sensor nodes to establish a common group key, while the latter allows pairs of nodes to share distinct keys.

Among the existing group key distribution schemes, self-healing group key distribution [9,21,28] particularly suits WSNs. A prominent property of this type of group key distribution is *self-healing*, which allows group members to recover lost group keys of past sessions based simply on the key update message of the current session. This makes group key distribution resilient to the lossy wireless channels of WSNs. Moreover, self-healing group key distribution offers group member *revocation* such that revoked group members can no longer get the group keys for new sessions after their revocation. This feature is extremely important in mitigating the effect of sensor node compromises: amputate compromised sensor nodes from the WSN, so that the adversary acquiring the secret information of the compromised nodes still cannot get the new group keys.

We observed that all the self-healing group key distribution schemes in the literature considered homogeneous WSNs where all sensor nodes are assumed to be of the same capabilities. However, homogeneous WSNs are not scalable. Indeed, both theoretical and empirical studies have found that the throughput of each sensor node decreases rapidly as the number of nodes increases, and as the traffic becomes heavy, the control overhead due to the underlying routing protocols will consume a large portion of the available bandwidth [11,13]. We are thus motivated to study self-healing group key distribution in *heterogenous* WSNs. A heterogenous WSN is composed of not only resource constrained sensor nodes, but also a number of more powerful high-end devices. More specifically, a WSN is partitioned into a number of *groups*/clusters, and a high-end device is placed into each group, acting as the *group manager*/cluster head. Compared to sensor nodes, a group manager is more powerful, and thus does not suffer from the resource scarceness problem as much as a sensor node does.

Our Contributions. Tailored to the heterogeneous WSN architecture, we propose the concept of *hierarchical* self-healing group key distribution. In particular, we formulate a security model for hierarchical self-healing group key distribution by revisiting and adapting Dutta *et al.*'s model [9]. We then propose a basic and an extended scheme, both proven secure under the model. Our construction basically follows Dutta *et al.*'s idea of a combination of a reverse and a forward one-way hash chain, but we show that their model and scheme have some weaknesses, which are rectified in ours. Our extended scheme further exploits the hierarchy of the heterogeneous architecture by secret-sharing the manger key of each group among all group managers, so as to counter possible compromises of group managers. To show that the (extended) scheme is efficient for WSNs, we implement the core (yet the most costly) element of the scheme upon MICAz mote [24], and the experiments demonstrate satisfactory performance.

2 Related Work

Key management and distribution is a security bootstrapping service, fundamental to many other security mechanisms in WSNs, hence tremendous effort has been dedicated to the study of this issue [7,9,8,12,18,19,20,21,27,31]. In general, public key cryptosystems are too expensive for WSNs, and symmetric key primitives such as secret key encryption or cryptographic hash function are often preferred. As such, key management and distribution in WSNs boils down to sharing of secret keys among sensor nodes. To achieve this objective, a commonly used approach is to pre-load a set of secrets inside sensor nodes before their deployment. These pre-loaded secrets are then used either directly as pairwise keys between each pair of sensor nodes, i.e., pair-wise key distribution [7,8,12,18,19,20,27,31], or as a basis to establish new common keys shared by a group of sensor nodes, i.e., group key distribution [6,9,15,20,21].

Among the existing group key distribution schemes, self-healing group key distribution is particularly suitable for WSNs, because of its self-healing and membership revocation properties. Staddon *et al.* [28] first proposed the concept and a concrete construction of self-healing group key distribution based on secret sharing of two dimensional polynomials. Their construction, however, is not efficient, suffering from high communication and storage overhead. Liu *et al.* [21] then generalized the security notions in [28], and presented a new scheme with better efficiency by combining personal secret distribution with the self-healing technique of [28]. Blundo *et al.* [4] analyzed the security definitions in [21,28] and concluded that it is impossible for any scheme to achieve all of the security requirements formulated in [21,28]. They then formulated a new definition for self-healing group key distribution and came up with a new scheme [5]. Blundo *et al.* [3] also showed an attack to the construction in [28] and discussed the use of randomness in self-healing group key distribution schemes. Other schemes based on the strategy in [28] include [25,14].

All the above self-healing group key distribution schemes are intended to achieve information theoretic security. In [9], Dutta *et al.* proposed novel computationally secure schemes, based on a combination of a reverse one-way hash chain and a forward one-way hash chain. While Dutta *et al.*'s model is weaker and cannot meet all the security requirements put forth in [21,28], their approach tremendously improves the efficiency of the information theoretically secure schemes. Unfortunately, as we shall show in Section 3, Dutta *et al.*'s definition on *the secrecy of personal secrets* in their model has some problems, and we give two attacks on the secrecy of personal secrets in their scheme. The schemes proposed by Du and He [10] followed Dutta *et al.*'s approach, and are also subject to our attacks. Besides rectifying the problems in Dutta *et al.*'s scheme, other differences between our schemes and Dutta *et al.*'s are as follows. First, our schemes are hierarchical, tailored to the heterogeneous WSNs. Second, our schemes achieve authenticated group key distribution, allowing every non-revoked sensor node to verify whether or not its generated group keys are valid, without requiring any extra communications.

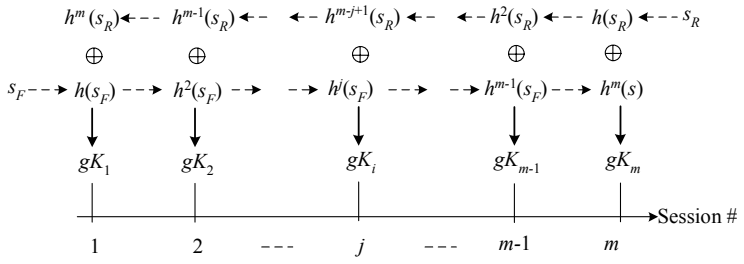

Fig. 1. Definition of Group Keys

3 Review and Analysis of Dutta *et al.*'s Scheme

3.1 Review of Dutta *et al.*'s Scheme

In Dutta *et al.*'s scheme [9], a WSN proceeds in sessions, and at the start of each session the base station broadcasts a key update message to the sensor nodes, enabling the latter to generate a new group key. Suppose the maximal number of sessions supported by the WSN is m. The group key for the j^{th} session is defined as $gK_j = h^j(s_F) + h^{m-j+1}(s_R)$, where h is a one-way hash function, and s_F, s_R are random seeds (see Figure 1). Here, $h^j(.) = \underbrace{h(h(\cdots h(.)))}_{j\ times}$. It can be seen that the hash chain associating with s_R is used in the *reverse* order thus called the reverse hash chain, and that associated with s_F called the forward hash chain. The details of Dutta *et al.*'s scheme are as follows.

- System Initialization. The base station chooses random seeds s_R and s_F, and it also selects m random t-degree polynomials $f_1(x), \cdots, f_m(x) \in F_q[x]$, each corresponding to a session, where F_q is a finite field with q being a large prime number, and t is a system parameter denoting the robustness of sensor nodes. The personal secret for a member sensor node i is defined to be $S_i = [f_1(i), \cdots, f_m(i)]$. Finally, the base station secretly sends S_i and s_F to each node i.
- Broadcast. At the start of each session, the base station broadcasts a key update message to enable sensor nodes to generate a new group key. Let $R_j = \{i_1, ..., i_w\}$ be the set of revoked sensor nodes upon the start of session $j \in \{1, \cdots, m\}$ and $|R_j| = w \leq t$. The base station computes the following polynomials:

$$\begin{aligned} r_j(x) &= (x - i_1) \cdots (x - i_w) \\ b_j(x) &= h(s_R)^{m-j+1}.r_j(x) + f_j(x), \end{aligned} \tag{1}$$

  where $r_j(x)$ is called the revocation polynomial. Finally, the base station broadcasts the key update message B_j to all sensor nodes, where $B_j = R_j \cup \{b_j(x)\}$.

– Session Key Generation. Upon receipt of B_j, if node i is not revoked, it is able to recover $h^{m-j+1}(s_R) = \frac{b_j(i)-f_j(i)}{r_j(i)}$. Then it continues to compute the group key $gK_j = h^{m-j+1}(s_R) + h^j(s_F)$ using s_F.
– Addition of New Group Member. A newly added member in session j is not allowed to compute group keys of previous sessions. To add a new member with ID α starting from session j, the group manager computes and gives $S_\alpha = \{f_j(\alpha), f_{j+1}(\alpha), \cdots, f_m(\alpha)\}$ and $h^j(s_F)$ to the node.

3.2 Attacks

In Dutta *et al.*'s model [9], the secrecy of personal secrets is defined as *any t or less revoked members cannot compute the personal secrets of other members.* We next give two attacks, showing that their scheme cannot achieve the secrecy of personal secrets. We also notice that the self-healing key distribution scheme proposed by Du and He in [10] follow Dutta *et al.*'s approach, and our attacks apply to their scheme too (to avoid repetition, we do not review their scheme).

Attack 1. In the above construction, the revocation polynomial $r_j(x)$ is simply defined as $r_j(x) = (x - i_1) \cdots (x - i_w)$. Let $f_j(x) = a_t x^t + \cdots + a_1 x + a_0$. It is clear that broadcasting $b_j(x)$ directly reveals $a_t, a_{t-1}, \cdots$, and a_{w+1}. This means that $f_j(x)$ only has $w + 1 \leq t$ unknown coefficients, i.e., $a_w, \cdots, a_0$, and any $w + 1$ (instead of $t + 1$) revoked nodes together can determine $f_j(x)$ and in turn compute $f_j(i)$ for any i. Therefore, the scheme cannot achieve the secrecy of personal secrets.

Attack 2. Let us consider a particular non-revoked node i in session j. From the broadcast message B_j, node i calculates $h(s_R)^{m-j+1} = \frac{b_j(i)-f_j(i)}{r_j(i)}$. Based on $h(s_R)^{m-j+1}$, node i can actually compute any $f(i'), i' \neq i$, as $f(i') = b_j(i') - h(s_R)^{m-j+1}.r_j(i')$. This suggests that once the group key for a session is established, the element of a sensor node's personal secret corresponding to that session is revealed to all other non-revoked nodes. As such, even node i is revoked in a subsequent session, it already knows a part (albeit corresponding to the past sessions) of other non-revoked nodes' personal secrets.

4 Model and Security Definition

4.1 Heterogeneous Architecture

We consider the heterogeneous architecture, where a WSN is partitioned into a number of *groups*. A high-end device is placed into each group, acting as the *group manager*. Compared to sensor nodes, high-end group managers have relatively higher computation capability, larger storage size, and longer radio range. They also have longer power supply, and can even be line-powered in some circumstances, e.g., when a WSN is deployed to monitor a building, the group managers can easily tap on the electricity lines to get power supply. Therefore

unlike sensor nodes, group managers do not suffer too much from the resource scarceness problem. Depending on applications, hardware capabilities of a group manager may vary from that comparable to a bluetooth device to that of a high end PDA. The introduction of high-end group managers into a WSN makes the once homogeneous network *heterogeneous*. The entire network including base station, group managers, and sensor nodes forms a logically hierarchical architecture, as depicted in Figure 2.

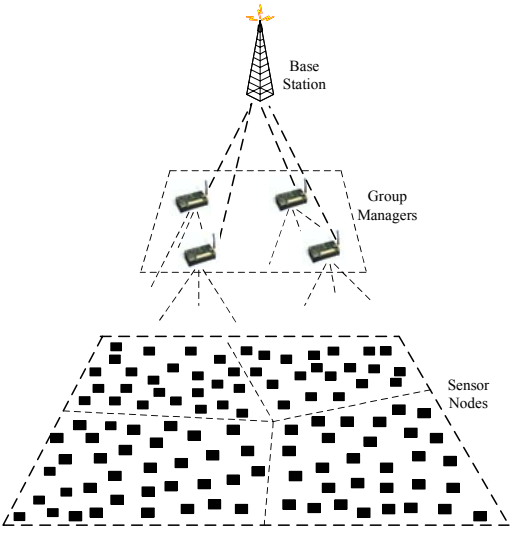

Fig. 2. Heterogeneous WSN Architecture

In this architecture, downlink messages broadcast by the base station directly reach sensor nodes, whereas uplink messages sent by a sensor node to the base stattion is forwarded via its group manager, which acts as an intermediary between the base station and the sensor nodes within its jurisdiction. A sensor node may reach the group manager directly, or by traversing a *short* multi-hop path. Intuitively, the inclusion of powerful group managers provides shortcuts for data delivered from the sensor nodes to the base station, so the overall system performance and in turn the lifetime of the network are expected to be greatly improved. Indeed, numerous studies have corroborated the higher efficiency of the heterogeneous WSN architecture, e.g., [16,26,29].

4.2 System Model

Three types of entities are involved in our hierarchical group key distribution system: base station, group managers, and a large number of sensor nodes. The sensor nodes are partitioned into a number of N_G groups, and each group G_ℓ has a group manager, $\ell \in \{1, \cdots, N_G\}$. Each sensor node in a group is uniquely identified by an ID number i, where $i \in I_\ell \subseteq \{1, \cdots, n\}$, where I_ℓ is the set of all node ID numbers in G_ℓ and n is the largest possible ID number in the system.

Corresponding to the heterogenous architecture, the keys held by the entities form a hierarchy, as shown in Figure 3: the base station holds a *root key* at level 2, each group manager has a distinct *manager key* at level 1, and at level 0 sensor nodes in each group hold a *group key* during each session. A key at a lower level is generated from the keys at higher levels, but not the other way around. This key hierarchy helps to implement "separation of duty" within the system, e.g., it is not necessary for the sensor nodes to process the control messages broadcast by the base station to the group managers.

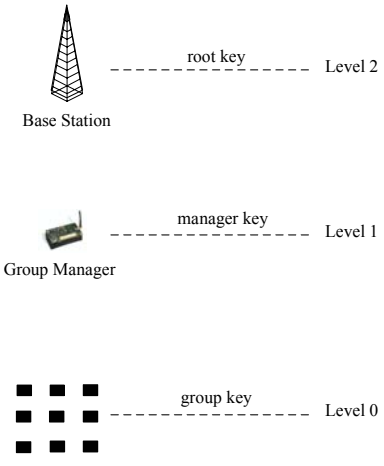

Fig. 3. Key Hiearchy

A group manager takes charge of distribution of group keys within its group. A group key is uniquely associated with a session. To distribute a group key for a new session, the group manager broadcasts a *key update message* to all its sensor nodes. The group key is then computed by a sensor node based on the received key update message and its preloaded *personal secret*. Denote the personal secret of sensor node i as S_i, which is a vector of m elements with m being the maximum number of sessions. Each element in S_i corresponds to a session and we use $S_i[j]$ to denote the element corresponding to the jth session, $j \in \{1, \cdots, m\}$. $S_i[j]$ becomes *obsolete* once the group key for the jth session is established; otherwise $S_i[j]$ is *fresh*. A sensor node can be revoked or non-revoked, and only non-revoked sensor nodes can compute the group keys.

4.3 Adversary Model

As usual, we assume that the base station is trusted. In our basic scheme, the group managers are also presumed trusted, but his assumption is removed in the extended scheme. We are mainly concerned with the distribution of group keys among sensor nodes. As such, we assume an adversary is able to passively eavesdrop on, or actively intercept, modify, insert, or drop key update messages

from a group manager to all its sensor nodes. We also allow the adversary to compromise up to t sensor nodes in a group, where t is a system parameter.

4.4 Security Definition

We formally define the concept and security requirements of hierarchical self-healing group key distribution, by revisiting and extending the definition in [9].

Definition 1. (Hierarchical Self-healing Group Key Distribution with t-Revocation) *Let N_G, n, m, t be system parameters defined as above. $\mathcal{D}$ is hierarchical self-healing group key distribution with t-revocation, if the following holds:*

a. (Key Hierarchy) *The manager keys held by the group managers are derived from the root key of the base station, but it is computationally infeasible to compute the root key from the manager keys. The same relationship should hold between group keys and the corresponding manager key.*
b. (Secrecy of Personal Secret) *For any $U_\ell \subset \{1, \cdots, n\}$ in group $G_\ell, \ell \in \{1, \cdots, N_G\}$, if $|U_\ell| \leq t$, then it is computationally infeasible for the nodes in U_ℓ to collectively determine the fresh elements of S_i for any $i \notin U_\ell$.*
c. (Authenticated Generation of Group Key) *Let $gK_{\ell,j}$ be the group key of group G_ℓ for session j, and $B_{\ell,j}$ be the broadcast key update message from the group manager, where $j \in \{1, \cdots, m\}$. For any non-revoked sensor node i in the group, $gK_{\ell,j}$ is efficiently computed from $B_{\ell,j}$ and $S_i[j]$ in an authenticated manner. On the contrary, it is computationally infeasible to compute $gK_{\ell,j}$ from the key update message or a personal secret alone.*
d. (t-Revocation) *For any session j, let $R_{\ell,j}$ be the set of revoked nodes in G_ℓ at the start of session j, where $|R_{\ell,j}| \leq t$, it is computationally infeasible to compute $gK_{\ell,j}$ from the broadcast message $B_{\ell,j}$ and $\{S_i\}_{i \in R_{\ell,j}}$.*
e. (t-wise Forward Secrecy) *Let $U_\ell \subseteq \{1, \cdots, n\}$ denote the sensor nodes which joined G_ℓ after session j. Given that $|U_\ell| \leq t$, it is computationally infeasible for all members in U_ℓ to collectively compute $gK_{\ell,1}, \cdots, gK_{\ell,j}$, even with the knowledge of $gK_{\ell,j+1}, \cdots, gK_{\ell,m}$.*
f. (Self-healing) *A non-revoked sensor node in G_ℓ between sessions j_1 and j_2, $1 \leq j_1 < j_2 \leq m$, can efficiently compute any $gK_{\ell,j}$, $j_1 \leq j \leq j_2$, from B_{ℓ,j_2} and its personal secret.*

Remark. Compared to Dutta *et al.*'s model, we distinguish between *obsolete elements* and *fresh elements* of a personal secret, and the secrecy of personal secret in our definition actually mandates the secrecy of fresh elements. This differentiation addresses our second attack, and suggests that personal secret should not be used for purposes other than distribution of group keys, and once an element is obsolete, it should be discarded immediately.

5 Basic Scheme

We first present a basic hierarchical self-healing group key distribution scheme, assuming that the group managers are trusted. We suppose that the set of

revoked users is monotonic, i.e., a revoked user never rejoins the network. Let F_q be a finite field, where q is a large prime number. All arithmetic operations below are performed in F_q. Let $h, h_R, h_F : \{0,1\}^* \rightarrow F_q$ be cryptographic hash functions, and N_G, n, m, t be system parameters. The basic scheme is a slight extension of Dutta *et al.*'s scheme reviewed above, and rectifies its weaknesses.

- **System Initialization.** The base station chooses a root key $rK = [rk_1, rk_2]$, where rk_1 and rk_2 are random numbers of appropriate length. For each group $G_\ell, \ell = \{1, \cdots, N_G\}$, the base station computes a manager key as $mK_\ell = [mk_{\ell,1}, mk_{\ell,2}]$, where $mk_{\ell,1} = h(G_\ell, rk_1)$ and $mk_{\ell,2} = h(G_\ell, rk_2)$. Clearly, it is computationally infeasible to compute the root key from the manager keys. Then the base station securely passes the manager keys to the corresponding group managers. We do not specify how this can be done, but it often suffices by using some out-of-band channel.
  Upon receipt of the manager keys, the group managers begin the preparation for setting up group keys. In particular, the group manager for G_ℓ whose manager key is $mK_\ell = [mk_{\ell,1}, mk_{\ell,2}]$ sets $mk_{\ell,1}$ to be the seed $s_{\ell,R}$ of the reverse one-way hash chain of length $m+1$:

$$\begin{aligned} k^j_{\ell,R} &= h_R(k^{j-1}_{\ell,R}) \\ &= h_R(h_R(k^{j-2}_{\ell,R})) = \cdots \\ &= h^j_R(s_{\ell,R}), 1 \le j \le m+1 \end{aligned} \tag{2}$$

  and sets $mk_{\ell,2}$ to be the seed $s_{\ell,F}$ for the forward hash chain of length m:

$$\begin{aligned} k^j_{\ell,F} &= h_F(k^{j-1}_{\ell,F}) \\ &= h_F(h_F(k^{j-2}_{\ell,F})) \cdots \\ &= h^j_F(s_{\ell,F}), 1 \le j \le m \end{aligned} \tag{3}$$

  The group key $gK_{\ell,j}$ for session $j \in \{1, \cdots, m\}$ is defined to be $gK_{\ell,j} = k^{m-j+1}_{\ell,R} + k^j_{\ell,F} = h^{m-j+1}_R(s_{\ell,R}) + h^j_F(s_{\ell,F})$. The group manager next selects m random t-degree polynomials $f_{\ell,1}(x), \cdots, f_{\ell,m}(x) \in F_q[x]$, each corresponding to a session. The personal secret for the member sensor node i is defined to be $S_i = [f_{\ell,1}(i), \cdots, f_{\ell,m}(i)]$. The group manager then sends $S_i, k^{m+1}_{\ell,R}$ and $s_{\ell,F}$ to each node i in a secure manner, e.g., preloading before the deployment of nodes. Note that $k^{m+1}_{\ell,R}$ will be used as the initial *authenticator*, denoted as Γ_ℓ, in subsequent group key generation process.
- **Broadcast.** The broadcast procedure for EACH group is almost the same as in Dutta *et al.*'s scheme, with the main exception on the computation of the revocation polynomial. In particular, let $R_{\ell,j} = \{i_1, ..., i_w\}$ be the set of revoked sensor nodes in G_ℓ upon the start of session $j \in \{1, \cdots, m\}$ and $|R_{\ell,j}| = w \le t$. The group manager chooses a random set $R'_{\ell,j} = \{i'_t, \cdots, i'_{w+1}\} \subset \{1, \cdots, n\} \setminus I_\ell$, where I_ℓ is the set of all node IDs in G_ℓ. That is, the group manager chooses $t - w$ random IDs that are not in that group. Then, the revocation polynomial $r_{\ell,j}(x)$ is computed as $r_{\ell,j}(x) =$

$(x-i_1)\cdots(x-i_w)(x-i'_{w+1})\cdots(x-i'_t)$. It is clear that $r_{\ell,j}(x)$ defined as such avoids our first attack. $b_{\ell,j}(x)$ is then computed with this $r_{\ell,j}(x)$ as in Dutta *et al.*'s scheme (Eqn. 1). Accordingly, the key update message $B_{\ell,j}$ also includes $R'_{\ell,j}$, i.e., $B_{\ell,j} = R_{\ell,j} \cup R'_{\ell,j} \cup \{b_{\ell,j}(x)\}$.

- **Session Key Generation.** The main difference from that in Dutta *et al.*'s scheme is that each sensor node in G_ℓ holds an authenticator Γ_ℓ. As such, when a non-revoked node recovers $k_{\ell,R}^{m-j+1} = \frac{b_{\ell,j}(i)-f_{\ell,j}(i)}{r_{\ell,j}(i)}$, it validates $k_{\ell,R}^{m-j+1}$ using Γ_ℓ: for example, if $\Gamma_\ell = k_{\ell,R}^{m+1}$ (the initial value)[1], then the validation is to test $\Gamma_\ell \stackrel{?}{=} h_R^j(k_{\ell,R}^{m-j+1})$. Other steps remain the same as in Dutta *et al.*'s scheme.
- **Addition of New Group Member.** This procedure is the same as in Dutta *et al.*'s scheme.

Efficiency. This scheme is highly efficient in terms of storage, communication, and computation overhead. For storage, the personal secret together with the authenticator accounts for $(m+1)\log q$ bits storage in each sensor node (compared to Dutta *et al.*'s scheme, ours only needs $\log q$-bit more storage for the authenticator). For communications, our scheme generates $t(\log q + \log n) \approx t \log q$ bits key update message (since $n \ll q$), which is almost the same as the bit length of the key update message in Dutta *et al.*'s scheme. For computation, no costly public key primitive is involved in our scheme, and the computation overhead inflicted upon sensor nodes includes only cryptographic hash function and polynomial operations.

Security. For security of the scheme, we have the following theorem and the proof can be found in [30].

Theorem 1. *The above construction is a hierarchical self-healing group key distribution scheme with respect to Definition* 1.

6 Extended Scheme

In the above basic scheme, we assumed that the group managers are trusted and not compromised. We next deal with the potential compromises of group managers in our system. There are two aspects to this problem: how to timely detect break-ins to group managers and how to mitigate the damage once a group manager is compromised. The first aspect can be addressed along the lines of cooperative intrusion detection techniques as proposed in [2,17,22]. We next discuss to address the second aspect, and in particular how to mitigate the damages caused by group managers' compromises.

It should be clear that once a group manager is compromised, the manager key it holds is unavoidably disclosed. In the basic scheme, since the manager key is used to derive all group session keys, once the manager key is revealed, so does all

[1] For better efficiency, a node should overwrite Γ_ℓ by setting $\Gamma_\ell = k_{\ell,R}^{m-j+1}$ at the end of the session key generation procedure.

the derived group keys. As such, probably the best we can expect is that *in case a group manager is compromised in a certain session, only the group keys for that and earlier sessions are revealed, without affecting subsequent sessions.* Note that since we desire self-healing, it is inevitable that disclosure of the manager key in a session leads to the disclosure of the group keys for all earlier sessions. To this end, the manager key of a group should be *sessional* too, rather than a constant quantity as in the basic scheme[2]. According to this rationale, we propose the following approaches to address the issue of compromises of group managers.

6.1 Naive Approach

A naive approach is to involve the base station into distributing the sessional manager keys to the group managers. Specifically, at the initialization phase, the base station generates $s_{\ell,R}$ and $s_{\ell,F}$ for each group G_ℓ using the root key, as per the basic scheme, i.e., $s_{\ell,R} = h(G_\ell, rk_1)$ and $s_{\ell,F} = h(G_\ell, rk_2)$. However, the base station does not pass $s_{\ell,R}$ to the group manager, but keeps it to itself. For each session $j \in \{1, \cdots, m\}$, the base station computes $k_{\ell,R}^{m-j} = h_R^{m-j}(s_{\ell,R})$ (Eqn. 2) and sends it to the corresponding group manager. Here we assume a secure communication channel between the base station and each group manager. Then, the sessional manager key held by the group manager is $mK_{\ell,j} = [k_{\ell,R}^{m-j}, s_{\ell,F}]$. Using $k_{\ell,R}^{m-j}$, the group manager computes and broadcasts $k_{\ell,R}^{m-j+1} = h_R(k_{\ell,R}^{m-j})$ to the sensor nodes as in the basic scheme (Eqn. 1), which enables the nodes to compute group key $gK_{\ell,j} = k_{\ell,R}^{m-j+1} + k_{\ell,F}^{j}$. It is easy to see that the manager keys thus generated sustain key hierarchy, and compromise of $mK_{\ell,j}$ does not disclose group keys for sessions later than $j+1$. This is *almost* the best we can expect.

This approach is quite simple. However, the involvement of the base station offsets the benefits offered by the heterogeneous architecture, one of which is to dispense with the implication of the base station into the management (including security enforcement) of individual groups. The involvement of the base station may result in single point of failure, in the sense that once the adversary manages to block the base station by, e.g., DoS attacks, the whole system is crashed.

6.2 An Extended Scheme

The extended scheme tries not to get the base station involved, as in the basic scheme. As such, secret-sharing $s_{\ell,R}$ of each group among multiple group

[2] One may argue that since the group manager is the only party that takes charge of the distribution of group session keys, once the group manager is compromised, all subsequent sessions of that group will fall to the control of the adversary, regardless of the measures taken to protect the manager key. This argument is actually based on the assumption that the adversary can continue controlling the compromised group manager. We, however, expect that the base station can timely detect the compromised group manager (e.g., using the cooperative intrusion detection techniques) and recover it in a certain later session. In fact, we believe that a WSN in real application should achieve this. Our extended schemes aim to work in such a scenario.

managers seems to be the only possible solution. The approach should satisfy that at no time should $s_{\ell,R}$ be reconstructed at any group manager including the one it belongs to. This requires computing $k^j_{\ell,R}, j = 1, \cdots, m$, in a distributed way, while without reconstructing $s_{\ell,R}$. It is clear that if $h_R()$ is a regular cryptographic one-way hash function, it is quite hard (if possible) to compute $k^j_{\ell,R} = h^j_R(s_{\ell,R})$ as required. We have to find a special $h_R()$ that facilitates the desired distributed computation.

Our choice for $h_R()$ is $h_R : Z_N \rightarrow QR_N$, and in particular $h_R(x) = x^2 \pmod N$, where N is a product of two large primes such that factorization of N is computationally intractable, and x's are sufficiently large numbers in Z_N. In such a case, $h_R(.)$ is a one-way function, under the intractability assumption of computing square root modulo a composite. The well known Rabin encryption is based on this assumption. $h_R(.)$ thus defined has the following property: suppose $s_{\ell,R}$ has t' multiplicative shares $\pi_1, \cdots, \pi_{t'}$ such that $s_{\ell,R} = \pi_1 \times \cdots \times \pi_{t'} \pmod N$, then $h^j_R(s_{\ell,R}) = (s_{\ell,R})^{2^j} = (\pi_1)^{2^j} \times \cdots \times (\pi_{t'})^{2^j} \pmod N$. This property well meets our need of computing $h^j_R(s_{\ell,R})$ without reconstructing $s_{\ell,R}$. By using this $h_R(.)$, we next highlight the main idea of our scheme. The communication channel among the group managers is assumed secure. Let us further suppose that we offer t'-robustness to our system, i.e., the adversary does not recover $s_{\ell,R}$ unless compromising t' or more group managers, where t' is a system parameter.

- **System Initialization.** The base station selects the root key and computes $s_{\ell,R}$ and $s_{\ell,F}$ for each group G_ℓ, as in the basic scheme. To secret-share $s_{\ell,R}$ among all group managers, the base station partitions $s_{\ell,R}$ into t' shares $\pi_1, \cdots, \pi_{t'}$ such that $s_{\ell,R} = \pi_1 \times \cdots \times \pi_{t'} \pmod N$; then securely sends π_1 together with $s_{\ell,F}$ to the group manager of G_ℓ, and sends each of the remaining shares to $\lfloor (N_G - 1)/(t' - 1) \rfloor$ other group managers. That is, the shares, except the one held by the group manager of G_ℓ, are evenly distributed among the remaining $N_G - 1$ group managers. Note that this offers resilience to the share availability, in the sense that loss of some shares does not affect computation of the manager keys. This also gives a higher weight to π_1 held by the group manager of G_ℓ, without which the group session keys cannot be computed.

  The steps taken by the group managers in preparation for setting up group keys are the same as in the basic scheme.
- **Broadcast.** In session j, the group manager of G_ℓ asks other group managers to help generate $k^{m-j}_{\ell,R} = h^{m-j}_R(s_{\ell,R})$ as follows: each group manager raises the share π_i at its disposal to the power of 2^{m-j}, i.e., $\zeta_i = \pi_i^{2^{m-j}}$, and passes the result to the group manager of G_ℓ, who then computes $k^{m-j}_{\ell,R}$ by pooling (multiplying) together a combination of appropriate ζ_i's (including its own). The pooling procedure has to filter out redundant and erroneous shares. We stress that there are many means to detect erroneous shares, e.g., in the initialization phase, the base station gives the group manager of G_ℓ an *authenticator* for each share. The manager key $mK_{\ell,j}$ is then set to

$mK_{\ell,j} = [k_{\ell,R}^{m-j}, s_F]$. The remaining steps are the same as in the above naive approach.

- **Session Key Generation & Addition of New Member.** These steps remain the same as in the basic scheme and the naive approach.

Security: Security of this approach is straightforward, given the security of the basic scheme. We only point out that in this approach, t-revocation is based on the intractability assumption of computing square root modulo a composite, instead of the one-way-ness of the cryptographic hash function.

Experimental Results: In both of the basic and the extended schemes, the bit length of q should be at least equal to that of $h_R(.)$, i.e., $|q| \geq |h_R(.)|$. For the basic scheme, since $h_R(.)$ can be instantiated by a regular cryptographic hash function, it suffices that $|q| = 161$, assuming $|h_R(.)| = 160$. However, in the extended scheme, $|h_R(.)| = |N|$. To make factorization of N hard, $|N|$ should be at least 1024 bits, so should be $|q|$. We thus need to examine the actual efficiency of the extended scheme.

For computation overhead, squaring operations (i.e., computing $h_R(.)$) dominate the workload of sensor nodes. Note that although a squaring operation is an exponentiation, it is essentially also a multiplication operation. Thus in principle, squaring operations are not deemed expensive. Nevertheless, as squaring operations are the most costly part in our scheme, it still makes sense to gauge their actual computation cost on real world sensors. To this end, we tested upon MICAz mote [24] running TinyOS 2.0. Hardware configuration of MICAz mote includes a ATMega 128L (8-bit,8MHz) processor, 128K-byte program memory, and 2.4 GHz radio transmission. Figure 4 lists the experimental results on the timing of squaring operations over x's of varying sizes, modulo a 1024-bit N. The

Size of x (bits)	Timing (millisecs)	Size of x (bits)	Timing (millisecs)
650	26	850	59.6
700	32.5	900	71
750	39.7	950	83.2
800	51.3	1000	98.6

Fig. 4. Experimental Results

results indicate that a squaring operation takes less than 100 milliseconds, which is quite satisfactory for WSNs. For storage overhead, each sensor node needs to store the personal secret, each element of which is $|q| = 1024$ bits. Recall that the MICAz mote we experimented upon has 128K-byte program memory. Even the mote allocates 8K-byte program memory to store personal secrets, the network can have approximately 60 sessions.

7 Conclusion

Both experimental and theoretical studies have shown that heterogeneous WSNs have better scalability and performance than homogenous ones. However, all existing self-healing group key distribution schemes consider homogenous WSNs. We were thus motivated to study hierarchical self-healing group key distribution for heterogenous WSNs. In particular, we formulated a model for hierarchical self-healing group key distribution, and proposed concrete schemes that achieve computational security (instead of information theoretic security as in previous schemes in the literature) and high efficiency.

Acknowledgement

This work is supported by the A*STAR project SEDS-0721330047.

References

1. Blundo, C.: Randomness in Self-healing Key Distribution Schemes. In: Proc. IEEE Information Theory Workshop, pp. 80–84 (2005)
2. Buchegger, S., Le Boudec, J.: Performance Analysis of the CONFIDANT Protocol Cooperation of Nodes - Fairness in Dynamic Ad-hoc Networks. In: Proc. 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2002, pp. 226–236 (2002)
3. Blundo, C., D'Arco, P., Listo, M.: A Flaw in A Self-healing Key Distribution Schemes. In: Proc. Information Theory Workshop, pp. 163–166 (2003)
4. Blundo, C., D'Arco, P., Santis, A., Listo, M.: Definitions and Bounds for Self-healing Key Distribution. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) ICALP 2004. LNCS, vol. 3142, pp. 234–245. Springer, Heidelberg (2004)
5. Blundo, C., D'Arco, P., Santis, A., Listo, M.: Design of Self-healing Key Distribution Schemes. Designs, Codes and Cryptography 32(1-3), 15–44 (2004)
6. Blundo, C., De Santis, A., Herzberg, A., Kutten, S., Vaccaro, U., Yung, M.: Perfectly-secure key distribution for dynamic conferences. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 471–486. Springer, Heidelberg (1993)
7. Chan, H., Perrig, A., Song, D.: Random Key Pre-distribution Schemes for Sensor Networks. In: Proc. IEEE Symposium on Security and Privacy, pp. 197–213 (2003)
8. Du, W.L., Deng, J., Han, Y.S., Varshney, P.K.: A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. In: Proc. ACM Conference on Computer and Communication Security, CCS 2003, pp. 42–51 (2003)
9. Dutta, R., Change, E.C., Mukhopadhyay, S.: Efficient Self-healing Key Distribution with Revocation for Wireless Sensor Networks Using One Way Key Chains. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 385–400. Springer, Heidelberg (2007)
10. Du, W., He, M.: Self-healing Key Distribution with Revocation and Resistance to the Collusion Attack in Wireless Sensor Networks. In: Baek, J., Bao, F., Chen, K., Lai, X. (eds.) ProvSec 2008. LNCS, vol. 5324, pp. 345–359. Springer, Heidelberg (2008)
11. Das, S., Perkins, C., Royer, E.: Performance Comparison of Two On-demand Routing Procotols for Ad Hoc Networks. In: Proc. of IEEE INFOCOM 2000, vol. 1, pp. 3–12. IEEE Press, Los Alamitos (2000)

12. Eschenauer, L., Gligor, V.D.: A Key-Management Scheme for Distributed Sensor Networks. In: Proc. ACM Conference on Computer and Communication Security, CCS 2002 (2002)
13. Gupta, P., Kumar, P.: The Capacity of Wireless Networks. IEEE Transactions on Information Theory 46(2), 388–404 (2000)
14. Hong, D., Kang, J.: An Efficient Key Distribution Scheme with Self-healing Property. IEEE Communication Letters 9, 759–761 (2005)
15. Huang, D., Mehta, M., Medhi, D., Harn, L.: Location-aware Key Management Scheme for Wireless Sensor Networks. In: Proc. 2nd ACM workshop on Security of Ad Hoc and Sensor Networks
16. http://www.intel.com/research/exploratory/heterogeneous.htm
17. Kachirski, O., Guha, R.: Effective Intrusion Detection Using Multiple Sensors in Wireless Ad Hoc Networks. In: Proc. 36th Annual Hawaii International Conference on System Sciences, HICSS 2003 (2003)
18. Liu, D., Ning, P.: Location-based Pairwise Key Establishement for Relatively Static Sensor Networks. In: Proc. ACM Workshop on Security of Ad hoc and Sensor Networks (2003)
19. Liu, D., Ning, P.: Improving Key Pre-distribution wih Deployment Knowledge in Static Sensor Networks. ACM Transactions on Sensor Networks (2005)
20. Liu, D., Ning, P., Du, W.L.: Group-based Key Pre-distribution in Wireless Sensor Networks. In: Proc. ACM Workshop on Wireless Security (2005)
21. Liu, D., Ning, P., Sun, K.: Efficient Self-Healing Group Key Distribution with revocation Capability. In: Proc. ACM Conference on Computer and Communication Security, CCS 2003 (2003)
22. Marchang, N., Datta, R.: Collaborative Techniques for Intrusion Detection in Mobile Ad-hoc Networks. Ad Hoc Network 6, 508–523 (2008)
23. Madden, S.R., Franklin, M.J., Hellerstein, J.M., Hong, W.: TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks. In: Proc. 5th Annul Symposium on Operating Systems Design and Implementation, OSDI 2002 (2002)
24. http://www.xbow.com/Products/productdetails.aspx?sid=156
25. More, S., Malkin, M., Staddon, J.: Sliding-window Self-healing Key Distribution with Revocation. In: Proc. ACM Workshop on Surviable and Self-regenerative System (2003)
26. Mache, J., Wan, C.Y., Yarvis, M.: Exploiting heterogeneity for sensor network security. In: Proc. IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON 2008, pp. 591–593 (2008)
27. Perrig, A., Szewczyk, R., Wen, V., Culler, D., Tygar, D.: SPINS: Security Protocols for Sensor Networks. Wireless Networks Journal, WINE (2002)
28. Staddon, J., Miner, S., Franklin, M., Balfanz, D., Malkin, M., Dean, D.: Self-healing Key Distribution with Revocation. In: Proc. IEEE Symposium on Security and Privacy, S&P 2002, pp. 241–257 (2002)
29. Yarvis, M., et al.: Exploiting Heterogeneity in Sensor Networks. In: Proc. IEEE INFOCOM 2005 (2005)
30. Yang, Y.J., Zhou, J.Y., Deng, R.H., Bao, F.: Hierarchical Self-Healing Key Distribution for Heterogeneous Wireless Sensor Networks. In: Proc. Securecomm 2009 (2009),
http://icsd.i2r.a-star.edu.sg/staff/yanjiang/papers/securecomm09.pdf
31. Zhu, S., Setia, S., Jajodia, S.: LEAP: Efficient Security Mechanisms for Large-scale Distributed Sensor Networks. In: Proc. ACM Conferenc on Computer and Communication Security, CCS 2003, pp. 62–72 (2003)