Optimal Construction of Unconditionally Secure ID-Based Key Sharing Scheme for Large-Scale Networks

Goichiro HANAOKA[1], Tsuyoshi NISHIOKA[1],
Yuliang ZHENG[2] and Hideki IMAI[1]

[1]The 3rd Department, Institute of Industrial Science, the University of Tokyo
7-22-1 Roppongi, Minato-ku, Tokyo 106-8558, JAPAN
Phone & Fax: +81-3-3402-7365
E-Mail: {hanaoka,nishioka}@imailab.iis.u-tokyo.ac.jp
imai@iis.u-tokyo.ac.jp
[2] School of Comp. & Info. Tech., Monash University,
McMahons Road, Frankston, VIC 3199, AUSTRALIA
E-Mail: yzheng@fcit.monash.edu.au

Optimal Construction of Unconditionally Secure ID-Based Key Sharing Scheme for Large-Scale Networks

Abstract

Efficient ID-based key sharing schemes are desired world-widely for secure communications on Internet and other networks. The Key Predistiribution Systems (KPS) are a large class of such key sharing schemes. The remarkable property of KPS is that in order to share the key, a participant should only input its partner's identifier to its secret KPS-algorithm. Although it has a lot of advantages in terms of efficiency, to achieve unconditional security a large amount of memory is required. While conventional KPS establishes communication links between any pair of entities in a communication system, in many practical communication systems such as broadcasting, not all links are required. In this article, we show the optimal method to remove these unnecessary communication links. In our scheme, the required memory for each entity is just proportional to the number of its partners, while that in conventional KPS is proportional to the number of entities in the whole system. As example, if an entity communicates only with $1/r$ of others, the required memory is reduced to be $1/r$ of that of conventional KPS. Furthermore, this memory size is proven to be optimal. Our scheme provides a more efficient way for secure communication especially in large-scale networks.

Key words: ID-based cryptosystem, KPS, collusion threshold, memory size

1 Introduction

For information security, ID-based key distribution technologies are quite important. The concept of ID-based key cryptosystems was originally proposed by Shamir[3, 4]. Maurer and Yacobi presented an ID-based key distribution scheme following Shamir's concept [5, 6]. However, their scheme requires a huge computational power. Okamoto and Tanaka[7] also proposed a key-distribution scheme based on a user's identifier, but it requires prior communications between a sender and a receiver to share their employed key. Although Tsujii and others proposed several ID-based key-distribution schemes[8, 9], almost all of them have been broken[10]. Thus, the performance of these schemes is unsatisfactory. However, Blom's ID-based key-distribution scheme[2], which is generalized by Matsumoto and Imai[1], has quite good properties in terms of computational complexity and non-interactivity. Many useful schemes based on Blom's scheme have been proposed[1, 11, 12, 13, 14, 15, 16, 17, 18, 19], and these are called Key Predistribution Systems (KPS).

In a KPS, no previous communication is required and its key-distribution procedure consists of simple calculations. Furthermore in order to share the key, a participant should only input its partner's identifier to its secret KPS-algorithm. Blundo et al.[14, 15, 16] showed a lower bound of memory size of users' secret algorithms and developed KPS for a conference-key distribution. Moreover Fiat and Naor[17], Kurosawa et al.[19] applied a KPS for a broadcasting encryption system.

Although KPS has many desired properties, it has also a problem: When a number of users, which exceeds a certain threshold, cooperate they can calculate the central authority's secret information. Thus, to achieve perfect security the collusion threshold is determined to be larger than the number of entities in the network. Setting up such a high collusion threshold in this scheme requires large amounts of memory in the center as well as for the users according to the collusion threshold. Solving this problem will make KPS much more attractive for ID-based key-distribution.

Although KPS provides common keys for all possible communication links among entities, in practical communication systems most of them are not necessary. By removing such unnecessary communication links, we can reduce the required memory significantly. In our scheme, the required memory for each entity is proportional to the number of its partners, while that in conventional KPS is proportional to the number of entities in the whole system. As example, if a entity communicates only with $1/r$ of others, the required memory is reduced to be $1/r$ of that of conventional KPS. Furthermore, this memory size is proven to be optimal. In this work, we also propose an optimal *asymmetric t-conference key distribution scheme*. Since this scheme has a good property, it is considered to be utilized effectively in other applications.

Section 2 gives a brief review of the KPS. Then in section 3, straight-forward implementation of our scheme and its problem are described. Section 4 explains asymmetric t-conference key scheme as the solution of the problem of the straight-forward implementation. This is followed by the evaluation and discussion of the security of our scheme in section 5. Section 6 closes the paper with some concluding remarks.

2 A Brief review of KPS

A KPS consists of two kinds of entities: One entity is the KPS center, the others are the users who want to share a common key. The KPS center possesses the KPS-center algorithm by which it can generate an individual secret algorithm for each user. These individual algorithms are (pre-) distributed by the center to their users and allow each user to calculate a common key from the ID of his communication partner. This section explains how the users' secret KPS-algorithms are generated and how users share a common key.

Let a symmetric function $G(x, y)$ be the KPS-center algorithm. Then, each entity u_i ($i = 1, 2, \cdots, N$) is given the secret algorithm $U_{u_i}(x)(= G(x, u_i))$ ($i = 1, 2, \cdots, N$), respectively. In order to share the communication key between u_i and u_j, they should simply input u_j and u_i to their secret algorithms, respectively. Since $G(x, y)$ is a symmetric function, both they obtain $k_{u_i u_j} = U_{u_i}(u_j) = U_{u_j}(u_i) = G(u_i, u_j)$.

KPS, has three noteworthy properties. First, there is no need to send messages for the key distribution between entities who want to establish a cryptographic communication channel. Second, its key-distribution procedure consists of simple calculations so that its computational costs are quite small. Finally, in order to share the key, a participant has only to input its partner's identifier to its secret algorithm. Thus, KPS is well applicable to one-pass or quick-response transactions, e.g. mail systems, broadcasting systems, electronic toll collection systems, and so on.

However, KPS has a certain collusion threshold; when more users cooperate they can calculate the KPS-center algorithm $G(x, y)$. Thus, to achieve perfect security the collusion threshold is determined to be larger than the number of entities in the network. Then, the required memory for users' secret algorithms are increased according to the collusion threshold. In the following subsection, the relationship between the memory size and the collusion threshold is discussed in more detail.

2.1 A lower bound of $|U_{u_i}(x)|$

For a random variable X, $H(X)$ denotes the entropy of X. Generally,

$$0 \leq H(X) \leq \log_2 |X|, \text{ where} X = \{x \mid \Pr(X = x) > 0\}. \quad (1)$$

In particular, $H(X) = \log_2 |X|$ iff X is uniformly distributed.

Blundo et al.[14] showed the lower bound of required memory size for users. Suppose that for each $P \subseteq \{u_1, u_2, \cdots, u_N\}$ such that $|P| = t$, there is a key k_P associated with P. And, each user $u_i \in P$ can compute k_P, and if $F \subseteq \{u_1, u_2, \cdots, u_N\}$,$|F| \leq \omega$ and $|F \cap P| = 0$, F cannot obtain any information about k_P. Then, the lower bound of the amount of users' secret algorithm U_{u_i} is estimated as follows:

$$\log_2 |U_{u_i}| \geq \begin{pmatrix} t+\omega-1 \\ t-1 \end{pmatrix} H(K), \tag{2}$$

where $k_P \in K$ for any P. In order to achieve perfect security, $\omega + t$ should be equivalent to the number of entities in the whole network. For $t = 2$, Eq. 2 becomes $\log_2 |U_{u_i}| \geq (\omega + 1)H(K)$.

2.2 Optimal schemes

Blundo et al.[14] presented a KPS which achieves the optimal memory size. In this scheme, the center chooses a random symmetric polynomial in t variables over $GF(q)$ in which the degree of any variable is at most ω, that is, a polynomial

$$f(x_1, \cdots, x_t) = \sum_{i_1=0}^{\omega} \cdots \sum_{i_t=0}^{\omega} a_{i_1 \cdots i_t} {x_1}^{i_1} \cdots {x_t}^{i_t}, \tag{3}$$

where $a_{i_1 \cdots i_t} = a_{\sigma(i_1 \cdots i_t)}$ for any permutation σ on $(i_1, \cdots, i_t)$. The center computes $U_{u_i} = f(u_i, x_2, \cdots, x_t)$ and gives U_{u_i} $(i = 1, \cdots, N)$ to u_i $(i = 1, \cdots, N)$, respectively. Then, they can share their communication keys by inputting $t-1$ partners' identifiers. For $t = 2$, Blom's scheme[2], Matsumoto-Imai scheme[1] and some others are also known as optimal schemes. Although these schemes achieve the optimal memory size: $\log_2 |U_{u_i}| = (\omega+1)H(K)$, the amount of memory is still large (For perfect security, ω must be equivalent to (the number of entities)-1). Especially, in large-scale networks required memory size is enlarged according to their high collusion threshold. Furthermore, on a smart card since its size of storage is strictly limited, the collusion threshold cannot be set up high enough to avoid strong collusion attacks by huge number of entities. "KPSL1 card"[20, 21], where the key length is 64bits. The secret-algorithm itself then consumes 64-KBytes of memory size in each IC card. Therefore KPS was considered to be somewhat expensive for real IC card systems at that time. By introducing 128~256bits symmetric key cryptosystems (namely, $H(K) = 128 \sim 256$bits), this problem will be more serious.

3 Straight-forward method for removing unnecessary functions

As already mentioned, some KPSs are proven to be optimal, and it is impossible to reduce the required memory size providing all the communication links. However, the required memory size by these schemes are still high. Although to reduce the collusion threshold or the key length is one possible solution to reduce the memory size, the security is also reduced considerably. Then, we pay attention to the unnecessary communication links in networks. By removing them, it is considered to be possible to reduce the memory size maintaining the same security level. In this section, we show a basic concept to attain the object and its requirements.

3.1 Unnecessary communication links

In a large-scale network, there are a lot of pairs of entities that do not communicate with each other at all. We consider mainly 2 reasons. Firstly, since to avoid illeagal use access controls are taken, some users are not allowed to access specific resources. Secondly, there exist many resources which perform only to some specific client computers. Also, pairs of entities, which is not related with each other, do not need to communicate with each other.

Although there are many unnecessary communication links, conventional KPSs cannot deal with them efficiently. Namely, in conventional KPSs it seems to be impossible to remove only unnecessary communication links.

3.2 Straight-forward implementation and its problem

A possible solution to reduce memory size by removing unnecessary communication links is to construct a whole large network by using small KPSs. Namely, if small KPSs are provided only for necessary communication links, only the unnecessary ones can be removed. However, straight-forward implementation of this approach has a serious problem. By this problem, the required memory can be even more than that of the conventional KPS. We explain this problem in more detail in followings.

To construct a large network by using small KPSs, we first divide the set of entities Υ to N_Υ subsets $\{\Upsilon_1, \Upsilon_2, \cdots, \Upsilon_{N_\Upsilon}\}$ $(|\Upsilon_i \cap \Upsilon_j| = 0)$. Each Υ_i $(i = 1, \cdots, N_\Upsilon)$ fullfills following condition: $\forall \upsilon_i \in \Upsilon_i$ communicates with $\forall \overline{\upsilon}_i \in \overline{\Upsilon}_i$, where $\overline{\Upsilon}_i = \{\Upsilon_j \mid j \in P_i\}$, $P_i \subseteq \{1, 2, \cdots, N_\Upsilon\}$, and for $i_1, i_2 \in \{1, 2, \cdots, N_\Upsilon\}$, if $i_1 \in P_{i_2}$, then $i_2 \in P_{i_1}$. For convenience, if $i_1 \in P_{i_2} (i_2 \in P_{i_1})$, we say $< i_1, i_2 >= 1$, otherwise, $< i_1, i_2 >= 0$.

Then, if a whole network is constructed by small KPSs straight-forwardly, critical problems are found in following 2 situations:

Case1: $< i_1, i_2 >= 1$, $< i_1, i_1 >= 0$ $(i_1 \neq i_2)$

For the communication $< i_1, i_2 >= 1$, a small KPS is provided. The collusion threshold of this KPS is determined to be equivalent to $|\Upsilon_{i_1}| + |\Upsilon_{i_2}| - 2$. From Eq.2, the required memory size for this communication is also proportional to $|\Upsilon_{i_1}| + |\Upsilon_{i_2}| - 2$. However, since $< i_1, i_1 >= 0$, even $\upsilon_{i_1} \in \Upsilon_{i_1}$ communicates only with $|\Upsilon_{i_2}|$ entities using such amount of memory.

Case2: $< i_1, i_2 >= 1$, $< i_1, i_3 >= 1$, $< i_2, i_3 >= 0$ $(i_1 \neq i_2, i_1 \neq i_3, i_2 \neq i_3)$

For these communications, we consider 2 kinds of construction of small KPSs: a KPS for $\{\Upsilon_{i_1}, \Upsilon_{i_2}\}$ and a KPS for $\{\Upsilon_{i_1}, \Upsilon_{i_3}\}$ are set up, or a KPS for $\{\Upsilon_{i_1}, \Upsilon_{i_2}, \Upsilon_{i_3}\}$ is set up. For the first construction, the collusion threshold of 2 KPSs are $|\Upsilon_{i_1}| + |\Upsilon_{i_2}| - 2$ and $|\Upsilon_{i_1}| + |\Upsilon_{i_3}| - 2$, respectively. Thus, the required memory size of $\upsilon_{i_1} \in \Upsilon_{i_1}$ for these communications is proportional to $2|\Upsilon_{i_1}| + |\Upsilon_{i_2}| + |\Upsilon_{i_3}| - 4$. On the other hand, for the second construction the collusion threshold of the KPS is $|\Upsilon_{i_1}| + |\Upsilon_{i_2}| + |\Upsilon_{i_3}| - 2$. Hence, the amount of memory of $\upsilon_{i_1} \in \Upsilon_{i_1}$ is proportional to $|\Upsilon_{i_1}| + |\Upsilon_{i_2}| + |\Upsilon_{i_3}| - 2$. Further, the required memory size for $\upsilon_{i_2} \in \Upsilon_{i_2}$ and $\upsilon_{i_3} \in \Upsilon_{i_3}$ are also proportional to $|\Upsilon_{i_1}| + |\Upsilon_{i_2}| + |\Upsilon_{i_3}| - 2$. Anyway, in both constructions since $\upsilon_{i_1} \in \Upsilon_{i_1}$ communicate only with $|\Upsilon_{i_2}| + |\Upsilon_{i_3}|$ entities by these KPS(s), the required memory size for υ_{i_1} is large comparing with the amount of memory. Moreover, in the second construction $\upsilon_{i_2} \in \Upsilon_{i_2}$ and $\upsilon_{i_3} \in \Upsilon_{i_3}$ communicate only with $|\Upsilon_{i_1}|$ entities. Hence, their required memory size is also large.

In the worst case, the required memory size for a entity is almost 2 times of that in conventional KPS. Namely, if $< i_1, i >= 1$ $(i = 1, \cdots, N_\Upsilon,\ i \neq i_1)$ and $< j, k >= 0$ $(j, k = 1, \cdots, N_\Upsilon,\ j, k \neq i_1)$, then, the required memory size is $\left(\sum_{i \in \{1,2,\cdots,N_\Upsilon\}, i \neq i_1} (|\Upsilon_i| + |\Upsilon_{i_1}| - 2)\right) H(K)$. On the other hand, in conventional KPS the required memory size is $\left(\sum_{i \in \{1,2,\cdots,N_\Upsilon\}, i \neq i_1} |\Upsilon_i| - 2\right) H(K)$.

Hence, straight-forward implementation of constructing a whole network by small KPSs is inefficient. Note that the **Case1** and **Case2** are not rare. These cases cannot be found iff Υ can be divided to hold the following condition:

$$< i, j >= \begin{cases} 0 & (i \neq j) \\ 1 & (i = j) \end{cases} \qquad \forall i, \forall j \in \{1, 2, \cdots, N_\Upsilon\}.$$

4 Optimal primitive

In order to construct a large-scale network by small KPSs efficiently, better primitives for **Case1** and **Case2** are required than the normal KPSs. In this section, the required property for the security primitive for **Case1** and **Case2** is discussed. Afterwards, we show an example which fulfills the requirements for **Case1** and **Case2**.

4.1 Generalization of Case1 and Case2

As already mentioned, **Case1** and **Case2** increase the memory size for entities. This problem is generalized by next **Lemma(∗)**.

Lemma(∗): *The required memory size for entities can be optimal if a whole network is constructed by normal KPSs and other primitives whose memory size for* $< i_1, i_2 >= 1, < i_1, i_1 >= 0, < i_2, i_2 >= 0$ *is optimal.*

Proof: When $< i_1, i_2 >= 1$ $(i_1 \neq i_2)$ is realized by a KPS, $< i_1, i_1 >= 1$ and $< i_2, i_2 >= 1$ are always realized simultaneously evenif they are not desired. In both **Case1** and **Case2**, such undesired functions bring the inefficiency. Thus, if an optimal primitive for $< i_1, i_2 >= 1, < i_1, i_1 >= 0, < i_2, i_2 >= 0$ is provided, **Case1** and **Case2** are dealt with optimally. If a function for communication $< i_1, i_1 >= 1$(or$< i_2, i_2 >= 1$) is required, a normal KPS is added as the optimal primitive for it. Hence, currently use of normal KPSs and optimal primitives for $< i_1, i_2 >= 1, < i_1, i_1 >= 0, < i_2, i_2 >= 0$ realizes optimal memory size for constructing large-scale networks by small key-sharing systems. □

4.2 Asymmetric t-conference key distribution

In this subsection, a lower bound of the memory size of a security primitive for $< i_1, i_2 >= 1, < i_1, i_1 >= 0, < i_2, i_2 >= 0$ $(i_1 \neq i_2)$ is shown. For other applications, we further generalize the primitive and call it *asymmetric t-conference key distribution.* Asymmetric t-conference key distribution is defined as follows:

Definition: *Let $\mathcal{U}$ be a set of entities and $\mathcal{U}$ is divided to t subsets $\mathcal{U} = \{\mathcal{U}_1, \mathcal{U}_2, \cdots, \mathcal{U}_t\}$. A key-sharing scheme for $\mathcal{U}$ is called asymmetric t-conference key distribution if*

1. *$\forall u_1 \in \mathcal{U}_1$, $\forall u_2 \in \mathcal{U}_2$, $\cdots$, $\forall u_t \in \mathcal{U}_t$ can compute their common key among them non-interactively (note that common keys among a same subset is not required).*
2. *Collusion thresholds $\psi_1, \psi_2, \cdots, \psi_t$ are independently set up for each of $\mathcal{U}_1, \mathcal{U}_2, \cdots, \mathcal{U}_t$. And unless a group of colluders $\mathcal{F}_i \in \mathcal{U}_i$ holds $|\mathcal{F}_i| > \psi_i$, any information of a common key is not exposed to entities who should not have it.*

Note that a security primitive for $< i_1, i_2 >= 1, < i_1, i_1 >= 0, < i_2, i_2 >= 0$ $(i_1 \neq i_2)$ is an asymmetric 2-conference key distribution.

Then, for $u_i \in \mathcal{U}_i$ the amount of memory of his secret algorithm U_i holds following lower bound:

$$\log_2 |U_i| \geq \left(\prod_{j \in \{1, \cdots, t\}, j \neq i} (\psi_j + 1) \right) H(K). \tag{4}$$

Proof: The mutual information between random valuables X and Y fulfills following 2 equations:

$$I(X;Y) = H(X) - H(X|Y), \tag{5}$$

$$I(X;Y) = I(Y;X). \tag{6}$$

¿From Eq.5 and Eq.6, following equation is obtained:

$$H(X) = H(Y) - H(Y|X) + H(X|Y). \tag{7}$$

Here, let $\mathcal{K}_{u_i}$ be a set of all common keys which are shared with $\forall u_1{}' \in \mathcal{U}_1{}' \subseteq \mathcal{U}_1$, $\forall u_2{}' \in \mathcal{U}_2{}' \subseteq \mathcal{U}_2$, $\cdots$, $\forall u_t{}' \in \mathcal{U}_t{}' \subseteq \mathcal{U}_t$ by asymmetric t-conference key distribution, where each $\mathcal{U}_i{}'$ $(i = 1, \cdots, t)$ holds $|\mathcal{U}_i{}'| = \psi_i$.

¿From Eq.7, the entropy of U_i is described as follows:

$$H(U_i) = H(\mathcal{K}_{u_i}) - H(\mathcal{K}_{u_i}|U_i) + H(U_i|\mathcal{K}_{u_i}). \tag{8}$$

Since in an asymmetric t-conference key distribution scheme, all of $\mathcal{K}_{u_i}$ can be computed by using U_i,

$$H(\mathcal{K}_{u_i}|U_i) = 0. \tag{9}$$

Then, from Eq.8 and Eq.9 following inequality is obtained:

$$H(U_{v_i}) = H(\mathcal{K}_{u_i}) + H(U_i|\mathcal{K}_{u_i}) \geq H(\mathcal{K}_{u_i}). \tag{10}$$

In a secure asymmetric t-conference key distribution system, mutual information between any pair of keys which belong to $\mathcal{K}_{u_i}$ must be 0. Thus, $H(\mathcal{K}_{u_i})$ holds

$$H(\mathcal{K}_{u_i}) = \sum_{K \in \mathcal{K}_{u_i}} H(K) = \left(\prod_{j \in \{1, \cdots, t\}, j \neq i} (\psi_j + 1) \right) H(K). \tag{11}$$

¿From Eq.10 and Eq.11, we obtain

$$H(U_i) \geq \left(\prod_{j \in \{1, \cdots, t\}, j \neq i} (\psi_j + 1) \right) H(K). \tag{12}$$

Hence, from Eq.1 Eq.12 becomes Eq.4. □

For unconditional security, each ψ_i $(i = 1, \cdots, t)$ should be equivalent to $|\mathcal{U}_i| - 1$. By introducing this collusion threshold, Eq.4 becomes

$$\log_2 |U_i| \geq \left(\prod_{j \in \{1, \cdots, t\}, j \neq i} |\mathcal{U}_j| \right) H(K). \tag{13}$$

4.3 An example of optimal asymmetric t-conference key distribution scheme

In this subsection an optimal asymmetric t-conference key distribution scheme is shown. In this scheme, the symmetric polynomial which is the KPS-center algorithm in Blundo et al.'s scheme is replaced with an asymmetric polynomial in t variables $x_1, x_2, \cdots, x_t$ over $GF(q)$ in which the degree of each variable is $\psi_1, \psi_2, \cdots, \psi_t$, respectively, that is, a polynomial

$$f(x_1, \cdots, x_t) = \sum_{i_1=0}^{\psi_1} \cdots \sum_{i_t=0}^{\psi_t} a_{i_1 \cdots i_t} {x_1}^{i_1} \cdots {x_t}^{i_t}. \tag{14}$$

Note that $a_{i_1 \cdots i_t}$ is not necessary to be equivalent to $a_{\sigma(i_1 \cdots i_t)}$ for any permutation σ on $(i_1, \cdots, i_t)$. The center computes $U_i = f(x_1, x_2, \cdots, x_t)|_{x_i = u_i}$ and gives each U_i to u_i, respectively. When u_i communicates with $u_1 \in \mathcal{U}_1, u_2 \in \mathcal{U}_2, \cdots, u_t \in \mathcal{U}_t$ (excluding $u_i \in \mathcal{U}_i$), u_i computes their communication keys by $U_i|_{x_1=u_1, x_2=u_2, \cdots, x_t=u_t \text{ (excluding } x_i=u_i)}$. By this procedure, an asymmetric t-conference key distribution is exactly realized.

In this scheme, the required memory size for U_i is estimated as follows:

$$\log_2 |U_i| = \left(\prod_{j \in \{1, \cdots, t\}, j \neq i} (\psi_j + 1) \right) H(K). \tag{15}$$

Thus, this scheme is optimum since we have already shown the lower bound on $|U_i|$ by Eq.4. For $t = 2$, we then obtain an optimal security primitive for $< i_1, i_2 >= 1, < i_1, i_1 >= 0, < i_2, i_2 >= 0$ $(i_1 \neq i_2)$. When we apply this, the required memory for $\upsilon_1 \in \Upsilon_1$ is $|\Upsilon_2| H(K)$, which is much less than that by normal KPSs as shown in 3.2.

5 Optimal construction by normal KPSs and asymmetric 2-conference key distribution schemes

As already mentioned in 4.1, if normal KPSs and asymmetric 2-conference key distribution schemes are applied, the required memory size can be optimal. In this section, we show an optimal construction of a large-scale network by removing unnecessary communication links.

Procedure of the center: The center first provides center algorithms of normal KPSs and asymmetric t-conference key distribution systems. For each $< i, i >= 1$ $(i = 1, 2, \cdots, N_\Upsilon)$, a normal KPS is applied. And for each $< i, j >= 1$ $(i \neq j)$, an optimal asymmetric 2-conference key distribution scheme is applied. $G^i(x, y)$ and $G^{ij}(x, y)$ $(i, j \in \{1, 2, \cdots, N_\Upsilon\},\ i \neq j)$ denotes the center algorithms for normal KPSs and asymmetric t-conference key distribution systems, respectively ($G^i(x, y)$ and $G^{ij}(x, y)$ holds $G^i(x, y) = G^i(y, x)$ and $G^{ij}(x, y) = G^{ji}(y, x)$, respectively). Then, the center gives the following secret algorithm U_{υ_i} to $\upsilon_i \in \Upsilon_i$:

$$U_{\upsilon_i} = \{ U_{\upsilon_i}^{ij} \ | \text{For} < i, j >= 1,\ U_{\upsilon_i}^{ij}(y) = G^i(\upsilon_i, y)\ (i = j),\ U_{\upsilon_i}^{ij}(y) = G^{ij}(\upsilon_i, y)\ (i \neq j) \} \tag{16}$$

Since applied KPSs and asymmetric 2-conference key distribution schemes are optimal, required memory size for each small key sharing system is estimated as follows:

$$\log_2 |U_{\upsilon_i}^{ij}(y)| = |\Upsilon_j| H(K). \tag{17}$$

As an optimal KPS, Blundo et al.'s scheme, Matsumoto-Imai scheme and Blom's scheme are available. On the other hand, as an optimal asymmetric 2-conference key distribution scheme, our scheme shown in 4.3 is available.

Procedure of entities: v_i computes the common key with $v_j \in \Upsilon_j$ as follows:

$$\begin{aligned} v_i : k_{v_i,v_j} &= U^{ij}_{v_i}(v_j), \\ v_j : k_{v_i,v_j} &= U^{ij}_{v_i}(v_j) = U^{ji}_{v_j}(v_i). \end{aligned} \tag{18}$$

6 Evaluation

6.1 Memory size for entities

By our construction, the required memory size for $v_i \in \Upsilon_i$ is estimated as follows:

$$\log_2 |U_{v_i}| = \left(\sum_{j \in \{1,\cdots,N_\Upsilon\}} < i,j > |\Upsilon_j| \right) H(K). \tag{19}$$

Here let $\log_2 |U'|$ be the required memory size for an entity when the a whole network is constructed only by one normal KPS. Then, we obtain following equation:

$$\log_2 |U_{v_i}| = \frac{\sum_{j \in \{1,\cdots,N_\Upsilon\}} (< i,j > |\Upsilon_j|)}{\left(\sum_{j \in \{1,\cdots,N_\Upsilon\}} |\Upsilon_j|\right) - 1} \log_2 |U'|. \tag{20}$$

Since $\sum_{j \in \{1,\cdots,N_\Upsilon\}} (< i,j > |\Upsilon_j|)$ is equivalent to the number of partners of v_i, and $\sum_{j \in \{1,\cdots,N_\Upsilon\}} |\Upsilon_j|$ is equivalent to the number of entities in the whole system, Eq.20 becomes

$$\log_2 |U_{v_i}| = \frac{\textbf{the number of partners}}{\textbf{the number of entities} - 1} \log_2 |U'|. \tag{21}$$

Namely, by using our scheme the memory size for an entity is reduced to be almost same as (# of partners)/(# of entities). Moreover, our scheme is optimal due to the discussion in Section 4.

6.2 Security

Our scheme is perfectly secure since any subset of entities have no information on a key they should not know. When we reduce the collusion threshold for reduction of memory size, the security becomes non-perfect. Namely, if the collusion threshold is less than the number of entities, by a collusion attack colluders can compute common keys of a victim. However, to success this attack a huge number of colluders is required and it seems still impossible in real world.

7 Conclusion

In this paper, an optimal construction of ID-based key sharing scheme for large-scale networks is proposed. It has been pointed out that to achieve perfect security huge amount of memory is required in conventional KPS, and it has been shown how KPS can be improved for practical communication systems. To be specific, by removing communication links that are not required in a practical communication system, the amount of memory is reduced significantly. IN our scheme, the required memory for each entity is (the number of partners) $\times$ (the length of a common key), while that in conventional KPS is (the number of entities) $\times$ (the length of a common key). As an example, if an entity communicate only with $1/r$ of others, the required memory is reduced to be almost $1/r$ of that of conventional KPS. Furthermore, our scheme is proven to be optimal. This makes our scheme attractive for various applications like broadcasting or E-commerce in the Internet. In this work, we also propose an optimal asymmetric t-conference key distribution scheme. Since this scheme has a good property, it is considered to be utilized effectively in other applications. Additionally, since public-key cryptosystems do not have advantages of KPS in terms of computational cost, ID-basedness, and so on, the efficient combination of a public-key cryptosystem and our scheme will realize a more efficient and secure communication system than one single use of a public-key cryptosystem.

References

[1] T. Matsumoto and H. Imai, "On the KEY PREDISTRIBUTION SYSTEM: A Practical Solution to the Key Distribution Problem," Proc. of CRYPTO'87, LNCS 293, Springer-Verlag, pp.185-193, 1987.

[2] R. Blom, "Non-public Key Distribution," Proc. of CRYPTO'82, Plenum Press, pp.231-236, 1983.

[3] A. Fiat and A. Shamir, "How to Prove Yourself: Practical Solutions to Identification and Signature Problems," Proc. of CRYPTO'86, LNCS 263, Springer-Verlag, pp.186-194, 1986

[4] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," Proc. of CRYPTO'84, LNCS 196, Springer-Verlag, pp.47-53, 1985.

[5] U. Maurer and Y. Yacobi, "Non-interactive Public-Key Cryptography," Proc. of Eurocrypt'91, LNCS 547, Springer-Verlag, pp.498-407, 1992.

[6] U. Maurer and Y. Yacobi, "A Remark on a Non-interactive Public-Key Distribution System," Proc. of Eurocrypt'92, LNCS 658, Springer-Verlag, pp.458-460, 1993.

[7] E. Okamoto and K. Tanaka, "Identity-Based Information Security management System for Personal Comuputer Networks," IEEE J. on Selected Areas in Commun., 7, 2, pp.290-294, 1989.

[8] H. Tanaka, "A Realization Scheme of the Identity-Based Cryptosystems," Proc. of CRYPTO'87, LNCS 293, Springer-Verlag, pp.340-349, 1988.

[9] S. Tsujii and J. Chao, "A New ID-Based Key Sharing System," Proc. of CRYPTO'91, LNCS 576, Springer-Verlag, pp.288-299, 1992.

[10] D. Coppersmith, "Attack on the Cryptographica Scheme NIKS-TAS," Proc. of CRYPTO'94, LNCS 839, Springer-Varlag, pp.40-49, 1994.

[11] L. Gong and D. J. Wheeler, "A Matrix Key-Distribution Scheme," Journal of Cryptology, vol. 2, pp.51-59, Springer-Verlag, 1993.

[12] W. A. Jackson, K. M. Martin, and C. M. O'Keefe, "Multisecret Threshold Schemes," Proc. of CRYPTO'93, LNCS 773, pp.126-135, Springer-Verlag, 1994.

[13] Y. Desmedt and V. Viswanathan, "Unconditionally Secure Dynamic Conference Key Distribution," IEEE, ISIT'98, 1998.

[14] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro and M. Yung, "Perfectly Secure Key Distribution for Dynamic Conferences," Proc. of CRYPTO '92, LNCS 740, Springer-Verlag, pp.471-486, 1993.

[15] C. Blundo, L.A. Frota Mattos and D.R. Stinson, "Trade-offs between Communication and Strage in Unconditionally Secure Schemes for Broadcast Encryption and Interactive Key Distribution," Proc. of CRYPTO '96, LNCS 1109, Springer-Verlag, pp.387-400, 1996.

[16] C. Blundo and A. Cresti, "Space Requirements for Broadcast Encryption," Proc. of Eurocrypt '94, LNCS 950, Springer-Verlag, pp.287-298, 1995.

[17] A. Fiat and M. Naor, "Broadcast Encryption," Proc. of CRYPTO '93, LNCS 773, Springer-Verlag, pp.480-491, 1994.

[18] K. Kurosawa, K. Okada and H. Saido, "New Combimatorial Bounds for Authentication Codes and Key Predistribution Schemes," Designs, Codes and Cryptography, 15, pp.87-100, 1998.

[19] K. Kurosawa, T. Yoshida, Y. Desmedt and M. Burmester, "Some Bounds and a Construction for Secure Broadcast Encryption," Proc. of ASIACRYPT '98, LNCS 1514, Springer-Verlag, pp.420-433, 1998.

[20] T. Matsumoto, Y. Takashima, H. Imai, M. Sasaki, H. Yoshikawa, and S. Watanabe, "A Prototype KPS and Its Application - IC Card Based Key Sharing and Cryptographic Communication -," Trans. of IEICE Vol. E 73, No. 7, July 1990, pp.1111-1119, 1990.

[21] T. Matsumoto, Y. Takashima, H. Imai, M. Sasaki, H. Yoshikawa, and S. Watanabe, "THE KPS CARD, IC Card for Cryptographic Communication Based on the Key Predistribution System," Proc. of SMART CARD 2000, IC Cards and Applications, Today and Tomorrow, Amsterdam, Oct., 1989.