An Efficient Hierarchical Identity-Based Key-Sharing Method Resistant against Collusion-Attacks[⋆]

Goichiro Hanaoka[1⋆⋆], Tsuyoshi Nishioka[2], Yuliang Zheng[3] 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@imailab.iis.u-tokyo.ac.jp
imai@iis.u-tokyo.ac.jp

[2] Information Technology R&D Center, Mitsubishi Electric Corporation
5-1-1 Ofuna, Kamakura, 247-8501, JAPAN
Phone: +81-467-41-2181 & Fax: +81-467-41-2185
E-Mail:nishioka@iss.isl.melco.co.jp

[3] The Peninsula School of Computing and Information Technology
Monash University, McMahons Road, Frankston
Melbourne, VIC 3199, Australia
Email: `yzheng@fcit.monash.edu.au`
URL: `http://www-pscit.fcit.monash.edu.au/~yuliang/`
Phone: +61 3 9904 4196, Fax: +61 3 9904 4124

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, on the other hand it is vulnerable by certain collusion attacks. 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 propose a new version of KPS which is called *Hierarchical KPS*. In Hierarchical KPS, simply by removing unnecessary communication links, we can significantly increase the collusion threshold. As an example, for a typical security parameter setting the collusion threshold of the Hierarchical KPS is 16 times higher than that of the conventional KPS while using the same amount of memory at the KPS center. The memory required by the user is even reduced for a factor 1/16 in comparison with the conventional linear scheme. Hence, Hierarchical KPS provides a more efficient method for secure communication.

[⋆] A part of this work was performed in part of Research for the Future Program (RFTF) supported by Japan Society for the Promotion of Science (JSPS) under contact no. JSPS-RETF 96P00604.

[⋆⋆] A Research Fellow of JSPS

© Springer-Verlag Berlin Heidelberg 1999

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 have 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 the 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], and known as 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], Kurosawa et al.[18] showed a lower bound of memory size of users' KPS-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, the following problem exists, as well: When a number of users, which exceeds a certain threshold, cooperate they can calculate the central authority's secret information. Setting up a higher collusion threshold in this scheme requires larger amounts of memory in the center as well as for the users. Solution of 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 increase the collusion threshold significantly. This will be explained by means of a new version of KPS called *Hierarchical KPS*. Hierarchical KPS demonstrates how to optimize a KPS for a communication system against collusion attacks. Hierarchical KPS is constructed based on the Matsumoto-Imai scheme[1]. Since the key-distribution procedure in the Matsumoto-Imai scheme consists of only simple calculations, computational cost in Hierarchical KPS is also quite small. As an example, for a typical security parameter setting, the collusion threshold of Hierarchical KPS is 16 times higher than that of the conventional KPS while using the same amount of memory in the KPS center. The memory required by the user is even reduced to 1/16 of that for the conventional linear scheme.

Section 2 gives a brief review of the KPS. Afterwards in section 3, Hierarchical KPS is introduced. This is followed by the evaluation and discussion of the security of Hierarchical KPS in section 4. Section 5 closes the paper with some concluding remarks.

2 A brief overview 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 a secret algorithm by which it can generate an individual KPS 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 in the manner of the Matsumoto-Imai scheme. Note that all the calculations in this article are related to the finite field GF(2).

Let the m-dimensional vectors x_A and x_B be the effective IDs of entities A and B, respectively. The $m \times m$ symmetric matrices $G^{(\mu)}$ $(\mu = 1, \cdots, h)$ are called KPS-center algorithm. The $G^{(\mu)}$s are produced by the KPS center and kept secret to all other entities. $G^{(\mu)}$ generates the μ-th bit of a communication key between users A and B, so h is the length of this key. $X_A^{(\mu)}$ and $X_B^{(\mu)}$ are the secret KPS-algorithms of A and B, respectively. $X_A^{(\mu)}$ and $X_B^{(\mu)}$ are calculated by the KPS center as follows:

$$X_A^{(\mu)} = x_A \; G^{(\mu)}, \tag{1}$$

$$X_B^{(\mu)} = x_B \; G^{(\mu)}. \tag{2}$$

$X_A^{(\mu)}$ and $X_B^{(\mu)}$ are contained in *tamper-resistant-modules* (TRM) and distributed to A and B, respectively. (If procedures for inputting data into TRM is thought to be complicated, TRM is not necessary.) By using $X_A^{(\mu)}$ and $X_B^{(\mu)}$, A and B share their symmetric key as follows:

$$A: \; k_{AB}^{(\mu)} = X_A^{(\mu)} \; {}^t x_B, \tag{3}$$

$$B: \; k_{AB}^{(\mu)} = X_B^{(\mu)} \; {}^t x_A, \tag{4}$$

where $k_{AB}^{(\mu)}$ indicates the μ-th bit of the shared key k_{AB} between A and B.

KPS, including the Matsumoto-Imai scheme, 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 KPS-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 center-algorithm $G^{(\mu)}$. For example in the Matsumoto-Imai scheme, as already mentioned above $G^{(\mu)}$ is a $m \times m$ matrix. Hence, by using m linearly independent secret KPS-algorithms, the KPS-center algorithm is easily revealed (note however that, in order to participate in this collusion attack, each adversary has to break his TRM). Thus, m is determined by the

Table 1. Required communications in practical communication systems, where ○, △ and × indicate required, partly required and unnecessary, respectively.

	consumer	provider	server
consumer	×	○	×
provider	○	△	△
server	×	△	△

number of users. In order to avoid such collusion attacks, we need to increase the value of m. However, since the number of elements of $G^{(\mu)}$ is m^2, a quite large memory size is required for the KPS center to increase the value of m. Further, the memory size of a user's secret KPS-algorithm is thereby enlarged in proportion to m. Although these memory sizes are not small, they are proven to be optimal[14]. Hence, in the conventional KPS, we cannot cope with collusion attacks efficiently. This can be a serious problem, especially in a situation where the available memory is strictly limited (e.g. IC cards). For example, $m = 8192$ is selected as the collusion threshold in "KPSL1 card"[20], 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. Furthermore by introducing 128∼256bits symmetric key cryptosystems, the required memory size will be 128∼256-KBytes.

Although the conventional KPS provides a common key between any pair of entities, most of them are not necessary in practical communication systems. When no keys are provided for such unnecessary communication links, the collusion threshold can be increased and the memory size of the users decreased, while the memory size of the KPS center stays the same.

3 Hierarchical KPS

In practical communication systems, such as broadcasting, entities are classified into 3 classes: *consumer, provider*, and *server*. Figure 1 displays the structure of their communication links. Consumers, i.e. the majority of the entities, receive information from any provider. Servers hold information needed by providers. For example, in broadcasting, addressees and broadcasting stations are regarded as consumers and providers, respectively. Certain entities that provide information for broadcasting stations are regarded as servers. In such a communication structure, communication links between consumers are not necessary. Only communication links to providers are required for the consumers. Similarly, although some communication links between providers and servers are required, not all of them are necessary. Furthermore, although communication links among providers/servers are required, not all of them are necessary. So, providers and servers can be divided into multiple groups. Then, we should realize the possibility to share a common key only for

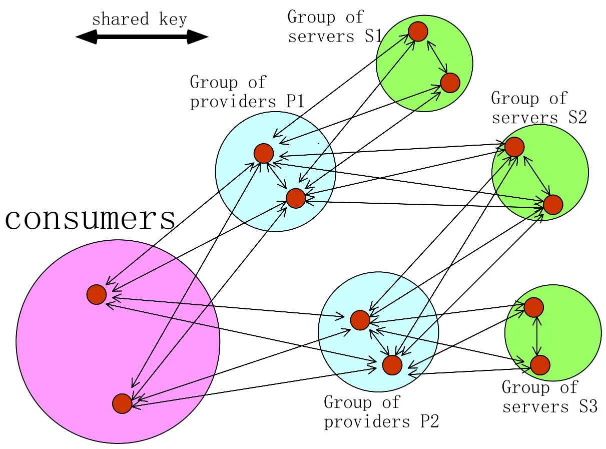


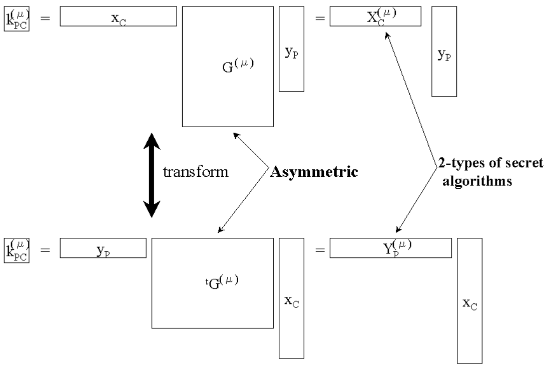
Fig. 1. Communication links in Hierarchical KPS.

- links between consumers and providers,
- links among providers who belong to same group,
- links among servers who belong to same group,
- links between providers and servers, supposed the group of providers and the group of servers are allowed to communicate with each other.

Necessary communication links in this structure are summarized in Table 1.

As mentioned above, in the Matsumoto-Imai scheme the collusion threshold can be increased by replacing the square $m \times m$ matrix $G^{(\mu)}$ of the center algorithm by a larger square matrix; this however requires a significantly larger memory size in the KPS center. Another possibility is to replace the $m \times m$ square matrix by a rectangular $m' \times n'$ matrix of the same size, $m^2 = m' \times n'$, $m' > m$, $n' < m$. This requires that the set of users is split into two distinct subsets. The threshold for a collusion of members of the first subset is m', and for a collusion of members of the second subset it is n'. Then a member of one subset can share a common key only with any member of the other subset; common keys between members of the same subset are not possible. This kind of KPS with asymmetric center algorithm will be used below to realize key distribution between consumers and providers, since no common keys are required among consumers.

From the requirement that the memory size of the KPS center should be fixed, i.e. from the equation $m^2 = m' \times n'$, it becomes clear that the collusion threshold m' for the consumers will increase when n' decreases. This means that there should be only few members in the second subset. Therefore a member of this subset is a group of providers who all provide access to several groups of servers. In other words, a “layer” of provider-groups is inserted between the consumers and the groups of servers, see Figure 1. Therefore the new version of



$k_{PC}^{(\mu)}$ = x_C $G^{(\mu)}$ y_P = $X_C^{(\mu)}$ y_P
transform
Asymmetric
2-types of secret algorithms
$k_{PC}^{(\mu)}$ = y_P ${}^tG^{(\mu)}$ x_C = $Y_P^{(\mu)}$ x_C

Fig. 2. Key distribution between a consumer and a provider in Hierarchical KPS.

KPS has been called "Hierarchical KPS". The following sections explain it in detail.

3.1 Key distribution between consumers and providers

Our improvement of KPS starts with replacing the symmetric matrices for the KPS center algorithm in the Matsumoto-Imai scheme by asymmetric $m \times n$ matrices $G^{(\mu)}$ $(\mu = 1, \cdots, h)$. Then key distribution between consumers and providers is implemented in the following way:

Let the m-dimensional vector x_C be the effective ID of consumer C, the n-dimensional vector y_P be the effective ID of provider P. Then C's secret KPS-algorithm $X_C^{(\mu)}$ is calculated by

$$X_C^{(\mu)} = x_C \; G^{(\mu)}, \tag{5}$$

and $Y_P^{(\mu)}$ is P's secret KPS-algorithm which is calculated as follows:

$$Y_P^{(\mu)} = y_P \; {}^tG^{(\mu)}. \tag{6}$$

C and P share their symmetric key k_{PC} according to

$$C: \; k_{PC}^{(\mu)} = x_C \; G^{(\mu)} \; {}^ty_P = X_C^{(\mu)} \; {}^ty_P, \tag{7}$$

$$P: \; k_{PC}^{(\mu)} = y_P \; {}^tG^{(\mu)} \; {}^tx_C = Y_P^{(\mu)} \; {}^tx_C, \tag{8}$$

where $k_{CP}^{(\mu)}$ indicates the μ-th bit of k_{CP}, the shared key between C and P. Figure 2 illustrates the key distribution between a consumer and a provider in Hierarchical KPS.

3.2 Key distribution among providers

In this subsection, we explain key distribution among providers with asymmetric matrices $G^{(\mu)}$.

For key distribution among providers we embed a symmetric matrix $G^{(\mu)}_{sym}$ into $G^{(\mu)}$, where $G^{(\mu)}_{sym}$ consists of rows in $G^{(\mu)}$ (Figure 3). By using $G^{(\mu)}_{sym}$, providers can share their keys as shown in Figure 4. According to the selection of rows belonging to $G^{(\mu)}_{sym}$ in $G^{(\mu)}$, elements from $Y^{(\mu)}_P$ and $Y^{(\mu)}_{P'}$ are selected to form n-dimensional vectors $\overline{Y^{(\mu)}_P}$ and $\overline{Y^{(\mu)}_{P'}}$. Using $\overline{Y^{(\mu)}_P}$ and $\overline{Y^{(\mu)}_{P'}}$, two providers P and P' share their key as follows:

$$P: \; k^{(\mu)}_{PP'} = \overline{Y^{(\mu)}_P} \; {}^t y_{P'}, \tag{9}$$

$$P': \; k^{(\mu)}_{PP'} = \overline{Y^{(\mu)}_{P'}} \; {}^t y_{P}, \tag{10}$$

Again, $k^{(\mu)}_{PP'}$ indicates the μ-th bit of the shared key $k_{PP'}$ between P and P'.

Although providers can share their keys by using this method, there are the following problems:

- $G^{(\mu)}_{sym}$ might be revealed by consumers' collusion attacks, if the selection of rows belonging to $G^{(\mu)}_{sym}$ in $G^{(\mu)}$ is exposed (for convenience, call this selection $k^{(\mu)}_{sel}$),
- A key between two providers cannot be longer than a key between a provider and a consumer.

As already mentioned, we assume that there are some groups of providers and that a provider communicates only with other providers in his group. The above problem can be solved if more than one $G^{(\mu)}_{sym}$ can be extracted from one $G^{(\mu)}$ and more than one $k^{(\mu)}_{sel}$ is distributed in each group of providers.

Suppose that $G^{(\mu),ij}_{sym}$ $(i = 1, \cdots, N_{sym}, \; j = 1, \cdots, N_P)$ are $n \times n$ symmetric matrices embedded in $G^{(\mu)}$, and $k^{(\mu),ij}_{sel}$ $(i = 1, \cdots, N_{sym}, \; j = 1, \cdots, N_P)$ are the

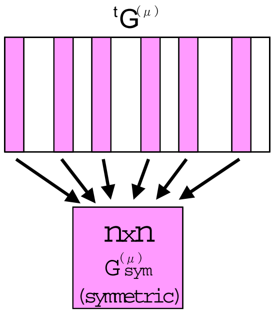
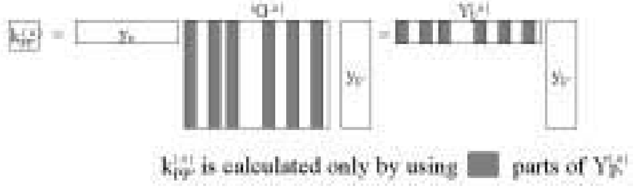


Fig. 3. Embedded symmetric matrix $G^{(\mu)}_{sym}$ in $G^{(\mu)}$.



$k^{(\mu)}_{PP'}$ =
y_P
${}^tG^{(\mu)}$
$y_{P'}$
=
$Y^{(\mu)}_P$
$y_{P'}$
$k^{(\mu)}_{PP'}$ is calculated only by using ■ parts of $Y^{(\mu)}_P$

Fig. 4. Key distribution among providers.

selection of rows belonging to $G_{sym}^{(\mu),ij}$ in $G^{(\mu)}$. N_{sym} is the number of embedded symmetric matrices that are distributed within one group of providers, and N_P is the number of groups of providers. Note that $N_{sym}N_P$ should be no more than m/n for the security of the system. $k_{sel}^{(\mu),ij}$ $(i = 1, \cdots, N_{sym})$ are distributed to all providers in the j-th group P_j. Then, key distribution between providers P and P', who both belong to P_j, is carried out as follows:

$$P: \; k_{PP'}^{(\mu),ij} = \overline{Y_P^{(\mu),ij}} \; {}^t y_{P'} \quad (i = 1, \cdots, N_{sym}), \tag{11}$$

$$P': \; k_{PP'}^{(\mu),ij} = \overline{Y_{P'}^{(\mu),ij}} \; {}^t y_{P} \quad (i = 1, \cdots, N_{sym}), \tag{12}$$

where $k_{PP'}^{(\mu),ij}$ $(i = 1, \cdots, N_{sym})$ indicates the μ-th bit of the shared key $k_{PP'}^{ij}$ $(i = 1, \cdots, N_{sym})$ between P and P', and elements from Y_P and $Y_{P'}$ are selected according to $k_{sel}^{(\mu),ij}$ $(i = 1, \cdots, N_{sym})$ to form n-dimensional vectors $\overline{Y_P^{(\mu),ij}}$ and $\overline{Y_{P'}^{(\mu),ij}}$ $(i = 1, \cdots, N_{sym})$.

So, if a $G_{sym}^{(\mu),i_0j}$ has been exposed by a certain consumers' attack, the providers in P_j can deal with this attack by using another $k_{sel}^{(\mu),i_1j}$, $i_1 \neq i_0$. Furthermore, by using multiple $k_{sel}^{(\mu)}$ simultaneously, providers can share longer keys. For example, if both $k_{sel}^{(\mu),i_0j}$ and $k_{sel}^{(\mu),i_1j}$ are used simultaneously, the length of the keys among providers in P_j can be $2h$, that is twice the length of the keys between consumers and providers. Accordingly, the keys shared among providers can be at most $N_{sym}h$.

Additionally, note that this scheme permits a provider to belong to multiple groups concurrently.

3.3 Key distribution between providers and servers

As already mentioned, servers can share keys with providers, assuming that the groups they belong to are allowed to communicate with each other. In this subsection, we show how to produce a server's secret KPS-algorithm.

Let the n-dimensional vectors z_S be the effective ID of server S and $Z_S^{(\mu),ij}$ be the secret KPS-algorithm of S which is calculated as follows:

$$Z_S^{(\mu),ij} = z_S \; G_{sym}^{(\mu),ij} \quad (i = 1, \cdots, N_{sym}), \tag{13}$$

Herein it is assumed that S belongs to group of servers S_j that is allowed to communicate with the providers in group P_j. By using this secret KPS- algorithm, communication keys are shared between S and P as follows:

$$S: \; k_{SP}^{(\mu),ij} = Z_S^{(\mu),ij} \; {}^t y_P \quad (i = 1, \cdots, N_{sym}), \tag{14}$$

$$P: \; k_{SP}^{(\mu),ij} = \overline{Y_P^{(\mu),ij}} \; {}^t z_S \quad (i = 1, \cdots, N_{sym}), \tag{15}$$

where $k_{SP}^{(\mu),ij}$ $(i = 1, \cdots, N_{sym})$ indicates the μ-th bit of the shared key k_{SP}^{ij} $(i = 1, \cdots, N_{sym})$ between S and P.

Similarly to the key distribution among providers, if $G_{sym}^{(\mu),i_0 j}$ is exposed by a certain attack, S and P can still share their key using other $Z_S^{(\mu),i_1 j}$ and $\overline{Y_P^{(\mu),i_1 j}}$, $i_1 \neq i_0$. And by concurrent use of their secret KPS-algorithms again longer keys can be used. For example, if $Z_S^{(\mu),i_0 j}, Z_S^{(\mu),i_1 j}$ and $\overline{Y_P^{(\mu),i_0 j}}, \overline{Y_P^{(\mu),i_1 j}}$ are used, the length of a shared key is $2h$. As above, the length of the key shared between providers and servers in this manner can be $N_{sym}h$ as the maximum.

Note that a group of servers can be allowed to communicate with multiple groups of providers in this way, and that a server can belong to multiple groups of servers.

3.4 Key distribution among servers

Any pair of servers in the same group can share their communication key using the servers' secret KPS-algorithms mentioned in **3.3**. Namely, a pair of servers S and S', who belong to S_j, share their common key as follows:

$$S: \; k_{SS'}^{(\mu),ij} = Z_S^{(\mu),ij} \; {}^t z_{S'} \quad (i = 1, \cdots, N_{sym}), \tag{16}$$

$$S': \; k_{SS'}^{(\mu),ij} = Z_{S'}^{(\mu),ij} \; {}^t z_S \quad (i = 1, \cdots, N_{sym}), \tag{17}$$

where $k_{SS'}^{(\mu),ij}$ $(i = 1, \cdots, N_{sym})$ indicates the μ-th bit of the shared key $k_{SS'}^{ij}$ $(i = 1, \cdots, N_{sym})$ between S and S', $z_{S'}$ is the effective ID of S', and $Z_{S'}^{(\mu),ij}$ $(i = 1, \cdots, N_{sym})$ are the secret KPS-algorithms of S' that are produced similarly to those of S.

Similarly to the key distribution among providers or that between providers and servers, if $G_{sym}^{(\mu),i_0 j}$ is exposed by a certain attack, S and S' can still share their key using other $Z_S^{(\mu),i_1 j}$ and $Z_{S'}^{(\mu),i_1 j}$. And concurrent use of their secret KPS-algorithms again results in longer keys. Using $Z_S^{(\mu),i_0 j}, Z_S^{(\mu),i_1 j}$ and $Z_{S'}^{(\mu),i_0 j}, Z_{S'}^{(\mu),i_1 j}$, the length of the shared key is doubled. By this, the keys shared among providers can be at most $N_{sym}h$ long.

4 Evaluation and security discussion

4.1 Communications with Hierarchical KPS

Here we confirm whether Hierarchical KPS can provides the required communications-links in practical communication systems or not. As already discussed, required communication-links are consumer-provider, provider-provider (within a group of providers), provider-server (if the group that the provider belongs to and the group that the server belongs to are allowed to communicate with each other), server-server (with in a group of servers). It can be seen that these communications are available by the method described in **3.1** $\sim$ **3.4**. Hence, it is confirmed that all required functions are provided by Hierarchical KPS.

Furthermore, Hierarchical KPS offers a higher level of security than the Matsumoto-Imai scheme. As mentioned in **3.2** $\sim$ **3.4**, the keys among providers,

Table 2. Collusion thresholds to calculate $G^{(\mu)}$,$G_{sym}^{(\mu),ij}$.

colluders	$G^{(\mu)}$	$G_{sym}^{(\mu),ij}$
providers	n	n
consumers	m	$m - n + \log_2 n$
servers	–	n†
providers + servers	n providers	n‡

† Collusion by servers that belong to group of servers S_j.
‡ Collusion by any providers and severs that belong to S_j.

Table 3. Required memory size for each type of entities.

	KPS center	consumer	provider	server
Hierarchical KPS	hnm	hn	hm	$hN_{sym}n$
conventional KPS	hnm	$h\sqrt{nm}$	$h\sqrt{nm}$	$h\sqrt{nm}$

those between providers and servers and those among servers can be $N_{sym}h$ bits long, what is more than the length h of keys between consumers and providers. Hence, these communications can be carried out more safely than those by the Matsumoto-Imai scheme, assuming that the number h of matrices for the KPS-center algorithm is the same in Hierarchical KPS and the Matsumoto-Imai scheme.

4.2 Collusion attack against $G^{(\mu)}$

There are mainly three kinds of collusion attacks against $G^{(\mu)}$: the consumers' collusion, the providers' collusion, and the mixed collusion of consumers and servers. The servers cannot reveal $G^{(\mu)}$ by themselves. Although servers and consumers can collude to reveal $G^{(\mu)}$, the influence of the servers in the attack is quite limited. Hence, attacks of the servers against $G^{(\mu)}$ are not regarded as a problem.

To break the whole system, a collusion of m consumers or n providers is needed from the view of information theory because the quantity of the center's secret information is hmn bits, while consumer's secret KPS-algorithm has hn bits information and provider's secret KPS-algorithm has hm bits information.

It should be noted that the mixed collusion between consumers and providers is inefficient since the informations available to consumers and to providers are not independent from each other. The number of either consumers or providers joining in the collusion attack must exceed the corresponding threshold m or n to succeed in the attack.

Actually, since in $G^{(\mu)}$ symmetric matrices are embedded, leakage of one $k_{sel}^{(\mu),ij}$ brings $\frac{n-1}{2}$ reduction of the collusion threshold of consumers. However,

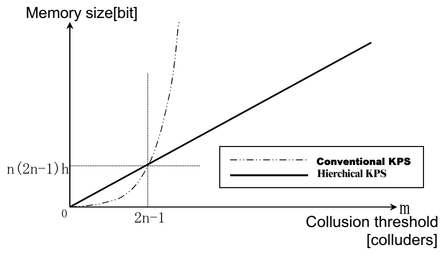


Fig. 5. Comparison of the required memory size for the KPS-center algorithm in Hierarchical KPS with that in conventional KPS, where n indicates the collusion threshold for providers.

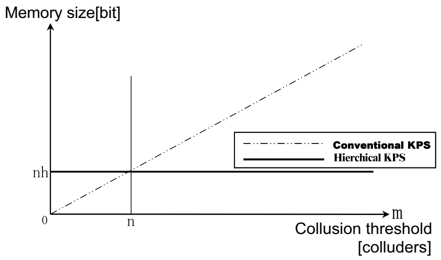


Fig. 6. Comparison of the required memory size for a consumer's secret KPS-algorithm in Hierarchical KPS with that in conventional KPS, where n indicates the collusion threshold for providers.

although all $k_{sel}^{(\mu),ij}$s are exposed, the collusion threshold is still high enough because $m \gg n$ (note that the collusion threshold of providers cannot be reduced). In Table 2, collusion thresholds against $G^{(\mu)}$ are shown. This means that a Hierarchical KPS can be designed as shown above based on the collusion thresholds n for consumers and m for providers. In conventional KPS, however mixed collusions can also be effective. This is why in conventional KPS the collusion threshold should be $(n+m)$, so that as center algorithm $(n+m) \times (n+m)$ matrices are needed in the Matsumoto-Imai scheme. Based on this assumption, the memory requirements of Hierarchical KPS and conventional KPS will be compared in the next section.

4.3 Memory requirements

Considering these collusion thresholds, m and n are determined mainly by the numbers of consumers and providers, respectively. Similarly, the required memory size for the KPS-center algorithm is determined to be proportional to n times m, while in the Matsumoto-Imai scheme the required memory size for the KPS-center algorithm is determined to be proportional to $(n+m)^2$. Furthermore, the memory size for the consumers' secret KPS-algorithms is proportional to n. Since in the Matsumoto-Imai scheme this is proportional to $(n+m)$, the memory size for the consumers' secret KPS-algorithms can be reduced considerably. Note that for general purpose applications the Matsumoto-Imai scheme achieves the optimal memory size for both the KPS-center and users like Blundo's scheme[14] and some others. Thus, we regard the memory size in the Matsumoto-Imai scheme as that of conventional KPS. As the number of consumers will usually be much higher than the number of providers, these reductions of memory size are quite significant. Figure 5 and Figure 6 show the memory size required for the KPS-center and a consumer. In the Matsumoto-Imai scheme, the required memory size for the KPS-center algorithm grows proportionally to the square of the col-

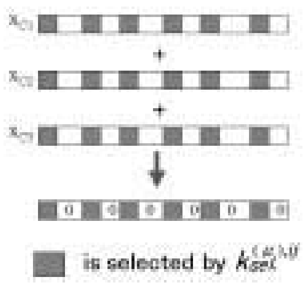


Fig. 7. The required combination of consumers' ID to reveal $G_{sym}^{(\mu),ij}$, where x_{C1}, x_{C2} and x_{C3} are effecitve identifiers of consumer $C1$, $C2$ and $C3$, respectively.

lusion threshold and the required memory size for users' secret KPS-algorithms proportionally to the collusion threshold. In contrast, in Hierarchical KPS the required memory size for the KPS-center algorithm increases proportionally to the collusion threshold for consumers, and the required memory size for consumers' secret KPS-algorithms remains unchanged while increasing the collusion threshold for consumers, assuming that the collusion threshold for providers is fixed (since the number of providers is much smaller than that of consumers, a quite low collusion threshold is sufficient to avoid a providers' collusion attack). Additionally, since the KPS-center algorithm of Hierarchical KPS consists of $N_{sym} \cdot N_P$ symmetric $n \times n$ matrices, we can reduce the required memory size for the KPS-center algorithm using their symmetrical property; in this way, the difference between the required memory size for the KPS-center algorithm in Hierarchical KPS and that for conventional KPS can be even more remarkable.

Also when taking into account the higher collusion threshold, the difference of the required memory size between Hierarchical KPS and conventional KPS is even more significant.

In summary, the collusion threshold in Hierarchical KPS can be much higher than that of conventional KPS by using same size of memory in the KPS center. Table 3 shows the required memory sizes for each type of entity. Only the memory size for the providers is larger in Hierarchical KPS than in the Matsumoto-Imai scheme. But this is not a serious problem since for providers such an amount of memory should be easily available.

4.4 Collusion attack against $G_{sym}^{(\mu),ij}$

Here, we especially discuss the collusion attack of consumers against $G_{sym}^{(\mu),ij}$ in more detail.

Note that in order to reveal $G_{sym}^{(\mu),ij}$, the adversary requires n combinations of the consumers' secret KPS-algorithms that fulfill the following condition:

Condition $(*)$ For the linear sum of the consumers' IDs participating in the combination, all the elements except those selected by $k_{sel}^{(\mu),ij}$ are entirely 0 (see Figure 7).

By using n such combinations, $G_{sym}^{(\mu),ij}$ can be revealed easily, when the involved sums are linearly independent. Hence, this attack can be realized by only n colluders in the worst case. However, the possibility of its success seems infeasible. Here, we estimate the number of colluders that yields feasible possibility to realize the attack.

When t consumers collude, the number of combinations of consumers' IDs is $2^t - 1$. Since the probability that a randomly selected combination fulfills the condition $(*)$ is 2^{n-m}, the expectation $E_{col}(t)$ of the number of the combinations that fulfill the condition $(*)$ is approximated as follows:

$$E_{col}(t) = (2^t - 1)(2^{n-m}) \simeq 2^{t+n-m}. \tag{18}$$

Thus, to achieve $E_{col}(t) \geq n$, we require $t \geq m - n + \log_2 n$. Hence, $m - n + \log_2 n$ can be regarded as the collusion threshold of this attack. Although this threshold seems to be still high enough, we can find that less than n colluders are required to reveal $G_{sym}^{(\mu),ij}$ if $k_{sel}^{(\mu),ij}$ is exposed. Thus, $k_{sel}^{(\mu),ij}$ must be kept secret to other entities besides its legal users. Basically, m and n were defined according to the number of consumers and providers, respectively. However, since the collusion threshold to reveal $G_{sym}^{(\mu),ij}$ by consumers is defined by both m and n, this must also be considered when choosing m and n. In The collusion thresholds against $G_{sym}^{(\mu),ij}$ is summarized Table 2. Although $k_{sel}^{(\mu),ij}$ can be revealed without difficulty if $G^{(\mu)}$ is exposed, we don't need to take care of this attack since the collusion threshold of $G^{(\mu)}$ is set up high enough to prevent any possible collusion attacks in real world.

As mentioned in **3.2**, by embedding multiple symmetric matrices in $G^{(\mu)}$, the damage of exposing $k_{sel}^{(\mu),ij}$ can be reduced. Namely, if a $G_{sym}^{(\mu),ij}$ is revealed, only the group that uses this $G_{sym}^{(\mu),ij}$ is affected. Although $G_{sym}^{(\mu),ij}$ is damaged, the communication can be realized by using another $G_{sym}^{(\mu),ij}$.

Additionally, although a collusion attack of providers can also reveal $G^{(\mu)}$, the collusion threshold of this attack is the same as that against $G_{sym}^{(\mu),ij}$ by the providers. Hence, in order to reveal $G_{sym}^{(\mu),ij}$, the providers have to reveal $G^{(\mu)}$. Besides, by a collusion attack of servers, $G_{sym}^{(\mu),ij}$ can be revealed. However, only the servers that belong to S_j can carry out this attack. In this attack, the collusion threshold is n, and it is regarded as high enough because there are not so many servers in comparison to consumers (although any provider can also participate in this collusion attack, this attack is still not serious).

4.5 Applications

Hierarchical KPS can be applied to quite many kinds of communication systems. In practical communication systems, we often find two kinds of entities that are regarded as consumers and providers. Usually, a minority of entities in the system communicates with almost all of the other entities, while the majority communicates only with specific entities (the minority). Hence, we can regard the minority and the majority as providers and consumers, respectively. Furthermore, in communication systems, we also often find entities that provide information to specific providers. Such entities are regarded as servers.

As an example, in broadcasting, addressees and broadcasting stations can be regarded as consumers and providers, respectively. Certain entities that serve information for the broadcasting stations take the role of servers. Assuming that the numbers of addressees, broadcasting stations and servers are 10,000,000, 5,000 and 200,000, respectively, we can set up $m = 131,072$ and $n = 512$ approximately. Then the number $N_{sym} \cdot N_P$ of embedded symmetric matrices is 256. Thus, the collusion threshold of addressees is $m = 131,072$, which is 16 times as large as the 8,192 with the conventional KPS, assuming that the utilized memory size is same in both Hierarchical KPS and the Matsumoto-Imai scheme. In this case, for the Matsumoto-Imai scheme 8192×8192 symmetric matrices are used as the KPS-center algorithm. Even when all the information for the location of embedded symmetric matrices in the center algorithm is exposed, the collusion threshold is still 8 times that of conventional KPS. Furthermore, Memory requirement (using h=64bits) is hn=32,768bits(=4-KBytes), which is 1/16 the requirement of 64-KBytes of the conventional KPS.

5 Conclusion

In this paper, Hierarchical KPS, which is a new style of KPS, has been proposed. It has been pointed out that certain collusion attacks can be effective against KPS, and on the other hand, it has been shown how KPS can be improved for practical communication systems to increase its resistance against collusion attacks. To be specific, by removing communication links that are not required in a practical communication system, resistance against collusion attacks is increased significantly. For a typical security parameter setting, the collusion threshold of the improved KPS is 16 times higher than that of the conventional KPS while using the same amount of memory in the KPS center. The memory required by the users is even reduced to be 1/16 of that for the conventional KPS. Hence, Hierarchical KPS provides a higher level of security against collusion attacks and a simplified implementation due to its reduced memory sizes. This makes Hierarchical KPS attractive for various applications like broadcasting or E-commerce in the Internet. 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.