# Cross-group Secret Sharing for Secure Cloud Storage Service

Ke, Chenyutao
Graduate School of Information Science and Electrical Engineering, Kyushu University

Anada, Hiroaki
Institute of Systems, Information Technologies and Nanotechnologies

Kawamoto, Junpei
Faculty of Information Science and Electrical Engineering, Kyushu University : Assistant Professor

Morozov, Kirill
Institute of Mathematics for Industry, Kyushu University : Assistant Professor

他

# Cross-group Secret Sharing for Secure Cloud Storage Service

Chenyutao Ke
Graduate School of Information Science and Electrical Engineering, Kyushu University,
744, Motooka, Nishi-ku,
Fukuoka, 819-0395 Japan
+81928023666
c.ka.521@s.kyushu-u.ac.jp

Hiroaki Anada
Institute of Systems, Information Technologies and Nanotechnologies
2-1-22, Momochihama, Sawara-ku,
Fukuoka, 814-0001 Japan
+81928523460
anada@isit.or.jp

Junpei Kawamoto
Faculty of Information Science and Electrical Engineering, Kyushu University,
744, Motooka, Nishi-ku,
Fukuoka, 819-0395 Japan
+81928023666
kawamoto@inf.kyushu-u.ac.jp

Kirill Morozov
Institute of Mathematics for Industry, Kyushu University,
744, Motooka, Nishi-ku,
Fukuoka, 819-0395 Japan
+81928024425
morozov@imi.kyushu-u.ac.jp

Kouichi Sakurai
Faculty of Information Science and Electrical Engineering, Kyushu University,
744, Motooka, Nishi-ku,
Fukuoka, 819-0395 Japan
+81928023666
sakurai@inf.kyushu-u.ac.jp

## ABSTRACT

With the spread of the Internet, many mobile devices are used in our daily lives, such as tablets and mobile phones. Then, personal data are often saved on data servers of the storage providers such as Amazon, Google, Yahoo, Baidu and others. In this context, secret sharing can be used to store personal data with several providers, simultaneously reducing the risk of data loss, the data leakage to unauthorized parties, and data falsification. Secret sharing is one of the solutions to combine security and availability in the distributed storage. However, few works considered servers' affiliations, and specifically, the problem that a malicious provider may recover secret data illegally through manipulation on servers that hold enough shares to recover the secret. In this paper, to resolve the problem, we propose a *two-threshold* secret sharing scheme in order to enforce a new type of cross-group policy. By combining $t$-out-of-$m$ providers' secret sharing scheme and a $k$-out-of-$n$ servers' secret sharing scheme via a one-way function or a one-time pad, we construct a scheme that forces $k$ shares to be collected from $m$ groups. Compared with previous work, our scheme can attain the functionalities of proactively updating shares and adding new shares with simple computation.

## Keywords

Storage Service, Shamir Secret Sharing, Cross-group Secret Sharing

## 1. INTRODUCTION

In recent years, with the development of high-performance servers and ultra-high-speed networks, the expansion of commercial cloud services has rapidly increased. Before such cloud services appeared, users had stored their data on their private PCs or other data storage devices. Currently, users can choose more options that are convenient; that is, on-line storage provided by cloud service providers, such as Amazon, Google, Yahoo, Baidu and others. If a user possesses large amount of data, it is convenient to use the cloud storage, because of its low cost. Other benefits of storing data on the cloud are that one can build an operating environment to support the business in a short time and can reduce the cost of initial investment. However, from the viewpoint of information security, certain risks must be mitigated such as unauthorized access, data leakage, data falsification and others. Thus, information management has become an important topic in the last few years.

Considering data security, we can backup data for $n$ copies and save those on different servers that are located in different regions. Even if $n - 1$ copies are destroyed, as long as one copy survives, the data remain available. However, this straightforward method will cause data leakage if only a single copy is stolen. Therefore, we require that the underlying protocol for backing up the data satisfies the following two properties at the same time.

1) A protocol can separate data into $n$ copies, and even if some of the copies are destroyed, the data remain available.
2) A protocol can separate data into $n$ copies, even if some of the copies are stolen, the data will not leak.

In fact, a technology that is called Secret Sharing can be used to solve this problem. In particular, Shamir's Secret Sharing Scheme [1] can separate data that we call "secret" into $n$ different shares, and if at least $k$ shares are collected, we can reconstruct secret. It means that even if $n - k$ shares are destroyed, we also can reconstruct the secret. Moreover, if $k - 1$ shares are stolen, the secret cannot leak to anybody.

Actually, commercial cloud data storage service based on secret sharing scheme were already introduced. For example, on May 18, 2015, NTT announced that it had developed an open source software Fast Secret Sharing Engine that was the first time in the world applied to the storage products "OpenStack Swift" of (OSS) "SHSS (Super High-speed Secret Sharing)". According to [22], SHSS realizes "in addition to data loss protection, high speed secret sharing engine that provides data encryption function".

Although secret sharing has already been used in commercial service, in previous work, there are mostly theoretical discussion and little consideration was given to actual problem about servers' affiliation. In other words, it was only suggested that the data should be distributed to different servers located in different regions, but not what servers should be selected. In fact, in previous work, one had not discussed about servers' affiliation, and a serious risk may be posed if the users distribute their data to only one provider's servers. If the provider which users' data are stored on was corrupted (which may happen, e.g. if computer virus infects the entire provider's network), an attacker can easily steal the users' data. In addition, a dishonest provider can recover the secret illegally as well.

In this work, we will design a secret sharing scheme in which the secret should be reconstructed only when the shares came from different cross-group providers. To enforce this policy, we introduce a new secret sharing scheme with two thresholds, where one threshold will be "responsible" for the number of providers involved in the reconstruction, and the other threshold – for the number of shares. We can say that our scheme fits infrastructure as a Service (IaaS).

## 1.1 Background and Motivation

The sensitive information, which we call a "secret", can be separated into $n$ shares by using Shamir's $k$-out-of-$n$ Secret Sharing Scheme, where $k$ is the threshold of recovering secret.

The reason why we separate a secret into $n$ shares is to reduce the risk of data loss and getting the ability of data recovery.

When distributing shares we have the following two points to pay attention to.

1) We must distribute $n$ shares to $m$ different providers.

It is obvious that considering the risk of information leakage, distributing $n$ shares to only one provider has higher risk of leaking information than distributing $n$ shares to $m$ different providers.

2) We must distribute $n$ shares to $l$ different regions.

Taking into account 2015 Tianjin explosions and 2011 Fukushima Daiichi nuclear disaster, the data which is stored on servers may be destroyed by serious disasters if all servers were located in the disaster site. Therefore, to ensure data recovery even if a part of shares is destroyed, we must distribute shares to $l$ different regions.

Here is a naïve method that we store each share at a different provider and a different region; that is, $n = m = l$. Such the naïve method forces each share to be distributed to different providers (and then we have to make contracts with each of them) but the number of providers is limited. Therefore, in practice, the number of shares is likely to be larger than the number of providers. This means that a provider may have more than one share, which, in turn, reduces the threshold in case of all her computers are compromised. Naturally, one would like to avoid such the disadvantage. In addition, in the real use-case, one may expect that that $m < l$, $l \leqslant n$. Hence in this paper, we will consider the situation where $m < n$.

Therefore, in the paper we had presented a two-threshold secret sharing scheme in order to enforce a new type of cross-group policy that can reduce the data leakage and the data loss risk. The central idea of our proposal is to generate two secrets: the data secret $s$ and the authority secret $v$. The data secret $s$ is the original data that we should protect and the authority secret $v$ is the mask used to control whether reconstruction is implement or not. In our proposal, we distribute disguised data shares $p_i$ instead of real data share $s_i$ to providers. In addition, the disguised data shares $p_i$ is generated from data shares $s_i$ and authority shares $v_i$. If the adversary compromises one provider, she could get more than $t$ disguised data shares $p_i$, but could not get more than one authority $v_i$. Therefore, she only can reconstruct disguised data secret $p$ and get authority share $v_i$, so that the data secret remains safe. The computational cost of our scheme will not exceed a double of that of Shamir scheme.

## 1.2 Previous Work

Smith et al. [18] proposed Layered Secret Sharing Scheme (Layered SSS). Layered SSS enables us to distribute shares hierarchically by treating a 1st level share as a 2nd level secret. The distributed shares are proposed to be stored in different regions.

Wang [19] and Martin [20] introduced a proper method to change threshold value when the number of shares was increased. Their method resolves the problem of the above reduction of security. In other words, in a $(t, n)$-threshold scheme, with the incrementing of shares, their methods can obtain a $(t', n')$-threshold scheme from a $(t, n)$-threshold scheme.

Beimel studied linear secret sharing schemes (LSSS) with general access structures [16]. However, when the parameters are growing, the share size of this scheme becomes prohibitively large.

Iftene and Boureanu proposed the Weighted Threshold Secret Sharing Scheme, which changed the secret reconstruction [21]. In the weighted threshold secret sharing schemes, the users do not have the same status. More exactly, a positive weight is associated to each user and the secret can be reconstructed if and only if the sum of the weights of all participants is greater than or equal to a fixed threshold.

## 1.3 Our Proposal and Comparison with the Related Works

Against the risk of information leakage by attackers that store all of shares on only one provider's servers and the risk of information loss by disasters, we have designed a proposal to reduce those risks. In our proposal, we have designed a client that is used to act as an agent to generate shares, to distribute shares and to reconstruct the secret. In other word, the client has two basic functions, "dealer function" and "reconstructer function".

Table 1. The difference between our proposal with previous work

| Name | Layered SS [18] | Linear SS [16] | Threshold-Changeable SS [19,20] | Our SS |
|---|---|---|---|---|
| Direct generation of shares | No | Yes | Yes | Yes |
| Direct reconstruction of secret | No | Yes | Yes | Yes |
| Renew shares | No | No | Yes | Yes |
| Add new shares | No | No | Yes | Yes |
| Need change threshold | - | - | Yes | No |

The dealer function is to generate shares and to distribute shares to different providers. In previous work, for one provider, with the increment of shares, the design policy of the threshold is reduced. That is, it will reduce security of data. To keep the security of secret sharing scheme, a proper method should be used to change threshold value dynamically. Nevertheless, in our proposal, even if one provider has more than one share: the design policy of the threshold is not reduced. That means the dealer function has the ability to protect the security of scheme even if the number of shares increased. In our proposal, a $(t, n)$-threshold scheme is replaced with a $(t, m)$-$(k, n)$-threshold scheme (i.e., $t$-out-of-$m$ scheme combined with $k$-out-of-$n$ shares threshold scheme). In the $(t, m)$-$(k, n)$-threshold scheme, we need not care about the number of shares which were increased afterward, that is different compared to Wang's proposal. Besides, the dealer function has an effective execution, due to simple low computation. Although Beimel's LSSS [16] also can be used to solve the problem that reducing the design policy of the threshold, its share size becomes prohibitively large.

The recoverer function is a process that reconstruct secret from distributed shares. When someone wants to reconstruct the secret, he should get over $k$ shares from at least $t$ providers. If the two conditions are not be satisfied, nobody can reconstruct the data secret by other methods. In other words, even if a provider has more than $k$ shares, without other providers' cooperation, he cannot perform secret reconstruction.

In order to protect against the share compromise, various methods have been proposed, such as to redistribute shares [3] [5]. In our proposal, we gave an effective method to update shares and revoke the leaked shares. Of course, a perfect online data storage service must have abilities to add new shares, and cheater detection [12] [13] [14]. Our proposal also has these features.

In Table 1, we show the comparison of our proposal with other Secret Sharing Schemes. Like Layered Secret Sharing Scheme, our proposal needs to use Shamir's secret sharing twice.

## 1.4 Our Contribution

We propose a new two-thresholds $(t, m)$-$(k, n)$ secret sharing scheme which permits the data owner to impose a condition that the reconstructing shares come from more than one storage provider. This can be useful, if the data owner prefers not to rely solely on one provider. Our proposal is easy to implement, and it incurs a little overhead compared to the ordinary Shamir scheme. Two variants are considered – the one with computational security based on one-way functions, and the one with information-theoretic security based on the one-time pad. In addition, we present the revocation protocol for compromised shares based on the proactive secret sharing scheme of Herzberg et al. [5] and the enrollment protocol for the new parties based on the protocol by Bao et al. [15].

Considering data security, we must distribute shares to different providers and distribute shares to different regions. Our proposal gave a simple and efficient method (i.e., $(t, m)$-$(k, n)$ secret sharing scheme) to achieve it.

## 2. PRELIMINARIES

### 2.1 Model and Assumptions.

We consider a cloud data storage consisted of $m$ groups and one dealer (In application, the dealer means client). One group means one service provider. In other words, we build a cloud data storage services by bundling $m$ existing cloud data storages. We denote each group by $P_i$ and the set of m groups by $A = \{P_1, P_2, \ldots, P_m\}$. We proactively share a secret $v$ in the groups with (k, m)-threshold secret sharing scheme, where $k$ is a given security parameter. Thus, $k - 1$ shares provide no information on the secret, while $k$ shares suffice for reconstruction of the secret. We name the secret $v$ the *authority secret*.

We assume each group has some data storage and in each of the $m$ groups, we assume we have $n$ servers. We denote each server by $Q_i$ and the set of servers by $B = \{Q_1, Q_2, \ldots, Q_n\}$. We also proactively share a secret $x$ in the servers with $(t, n)$-threshold secret sharing scheme, where $t$ is also a given security parameter. We name the secret $s$ data secret.

The goals of our scheme are to prevent adversaries from obtaining data secret $s$ and to prevent an adversarial group from reconstructing data secret $s$. In other words, anyone who wants to reconstruct secret $x$ must have $k$ groups' agreement and have $t$ shares. Someone who have $t$ shares, but have no more than $k - 1$ groups' agreement cannot reconstruct data secret $x$.

Fig. 1 shows an example of our data storage service, where the parties in the same group hold the same share of authority secret e.g. servers in group 1 have $v_1$, and all servers have different shares of data secret.

Each group in $A$ has some servers in $B$. We assume each group can access only own servers arbitrarily but cannot access other groups' servers. When a server in a group will be replaced, we assume all data on the old server will easily be moved to a new server. When a group wants to add a new server, we consider two cases: if the group already has $t$ shares, the group can easily use Shamir's Secret Sharing Scheme to build a new share; otherwise, if the group has less than $t$ shares, it needs to cooperate with other group to build a new share. Note that, in the latter case, if the groups' shares are not encrypted, the provider not only can use Shamir's Secret Sharing Scheme to build a new share, but also can reconstruct the secret, which is undesirable.

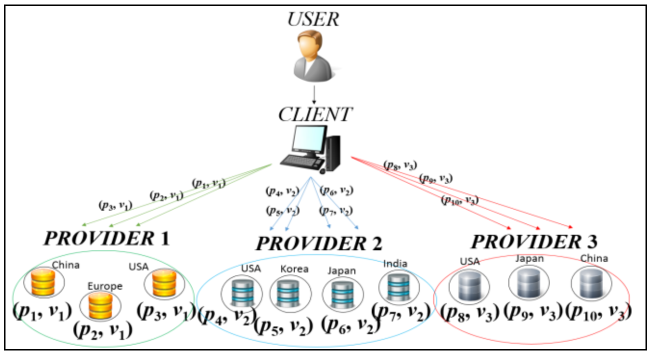

Fig. 1. Sample of proposal data storage service

(3 groups and 10 servers).

An adversary can corrupt any server or any group at any time. However, in a certain period, the adversary cannot corrupt more than $t - 1$ servers and get more than $t - 1$ shares.

## 2.2 Cryptographic Tools.

Our proposal is based on Shamir's secret sharing scheme. Throughout this paper, $p$ and $q$ denote large primes such that $q$ divides $p - 1$, $G_q$ is the unique subgroup of $Z^*_p$ of order $q$, and $g$ is a generator of $G_q$. It can easily be tested if an element $a \in Z_p$ is in $G_q$ since

$$a \in G_q \leftrightarrow a^q = 1.$$

As any element $b \neq 1$ in $G_q$ generates the group, the discrete logarithm of a $\in G_q$ with respect to the base $b$ is defined and it is denoted by $log_b(a)$.

For any integer $x$ the length of the binary representation of $x$ is denoted by $|x|$.

## 3. CROSS-GROUP SECRET SHARING WITH COMPUTATIONAL SECURITY

In the paper, we propose a two thresholds secret sharing scheme, $(t, m)$-$(k, n)$ secret sharing scheme. The two thresholds were based on Shamir's Secret Sharing Scheme. Only if the two thresholds were satisfied, data secret will be reconstructed correctly.

In this section, we present a solution of $(t, m)$-$(k, n)$ secret sharing scheme with computational security, i.e., we assume that the adversary is bound to run in probabilistic polynomial time.

## 3.1 Definition

In this section, we define the dealer as a client function, which realize distributing shares and recovering data secret $s$ from data shares. We utilize a one-way function that has a key. We denote the function by $f_h(x)$ where $h$ is the key. The data secret $s$ will be distributed as $n$ shares. We denote the i-th share of $s$ by $s_i$. The authority secret $v$ will also be distributed as $m$ shares. We denote the $i$-th share of $v$ by $v_i$.

We consider the difference between the data secret and the authority secret. $p$ denotes the difference, i.e. $p = s - q$. We distribute the difference to the servers. $p_i$ denotes the $i$-th share of $p$.

## 3.2 Our Scheme with Computational Security

Here, when someone wants to reconstruct the data secret $s$, as must necessarily participate in the threshold number group, was stablished the authority secret $v$ is an access prerequisite to data secret $s$. We use authority secret $v$ to make shares of $v_i$ and distribute to every group. Of course, when the authority secret share $v_i$ is stored on each group, there is a risk that an attacker may steal the share of $v_i$ and reconstruct authority secret $v$. To prevent it, there is a need for encrypting the authority secret $v$. In this paper, we want to encrypt the authority secret $v$ by using one-way-function that has a secret key $h$. Indeed, not only we can encrypt the authority secret $v$, but also we can encrypt data secret $s$. However, in the latter case, it is becomes hard to enroll new parties and thus we encrypt authority secret $v$ instead of encrypt data secret $s$.

***Secret Sharing***

1. Dealer generate a key $h$ at random whose length can be set appropriately.
2. Dealer calculates $q$ follow the below formula.

$$q = f_h(v).$$

   Here, $f_h(x)$ is a one-way-function, which has a key of $h$.
3. Dealer calculates disguised data secret $p$ as follows

$$p = s - q.$$

4. Dealer creates disguised data shares $p_i$ and $v_j$ by using Shamir secret sharing scheme.
   Here, $p_i$ is share of $p$, $v_j$ is share of $v$.
5. Dealer distributes pairs $(p_i, v_j)$ to corresponding groups.

When a party called Recoverer wants to reconstruct data secret $s$, firstly, she should have the right to access authority secret $v$. In order to access $v$, she should reconstruct $q$ and (she should obtain $h$ from the Dealer) to access $v$.

***Secret Reconstruction***

1. Recoverer gets pairs $(p_i, v_j)$ from different groups.
2. Recoverer reconstruct $p$ by using $p_i$. (The number of $p_i$ must over $t$).
3. Recoverer reconstruct $q$ by using $q_i$. (The number of $q_i$ must over $k$).
4. Recoverer uses $v_j$ to recover authority secret $v$.
5. Recoverer uses one way-function which has a key $h$ to calculate $q$ like below.

$$q = f_h(v)$$

6. Finally, Recoverer reconstruct data secret $s$ by using above formula.

$$s = p + q$$

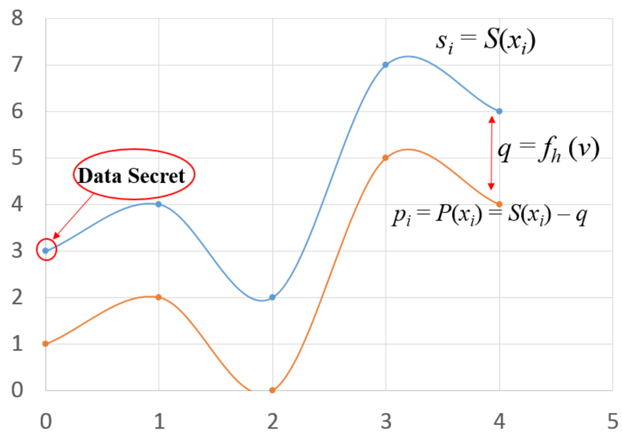

Fig. 2. The illustration of Lagrange polynomials with an auxiliary key

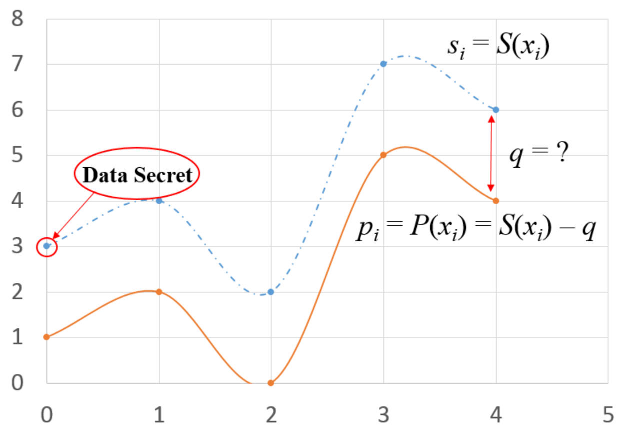

Fig. 3. The illustration of Lagrange polynomials without an auxiliary key

**Table 2.** Show the definition of our proposal in session 3.

| Name | Definitions |
|---|---|
| Dealer | Client function which makes shares and distributes it |
| Recoverer | Client function which reconstructs the secret from shares |
| Group | It refers to one of the providers |
| Server | The location where shares are stored |
| $s$ | Data secret |
| $v$ | Authority secret |
| $f_h(x)$ | $f_h(x)$ is a one-way-function which has the key of $h$ |
| $s_i$ | Shares of the data secret $s$ |
| $v_i$ | Shares of Authority secret $v$ |
| $P$ | $p = s - q$ |
| $p_i$ | Shares of $p$ |

***Analysis of our proposal in real application***

Fig. 2 shows the relation of $p_i$ and $s_i$. In our proposal, the most important thing is protecting the key of $h$, which means the key of $h$ has the highest security level in our proposal. As we know, in the online storage products, users may login to their accounts from any compute. But their data secret showed in Fig. 2 (marked by the red circle) can be reconstructed by using the functions $S(x_i)$ or $P(x_i)$. To reconstruct $S(x_i)$ or $P(x_i)$, which are the Lagrange polynomials, one will use the polynomial interpolation. In our proposal, the pairs $(p_i, v_j)$ can be used to reconstruct two Lagrange polynomials and then also the data secret $p$ and the authority secret $v$. However, if one does not hold $h$, one cannot compute the result $q = f_h(v)$, and hence cannot reconstruct the data secret $s$ by using $s = p + q$. This point is illustrated on Fig. 3.

In the Fig. 2, the data secret $s = S(0)$ is masked by using $q = f_h(v)$. We do not use $p = s - v$ to mask the data secret $s$ directly, because the authority share $v_i$ may be reused to mask different data secret. Moreover, the authority share may leak to the adversary. Therefore, if we do not use one-way function, which has a key of $h$, the adversary may reconstruct data secret $s$ easily.

The $h$ can be seen as special key for encrypting special personal data that should be safely stored by its users, preferably on a special secure hardware.

## 4. CROSS-GROUP SECRET SHARING WITH INFORMATION-THEORETIC SECURITY

In section 3, the scheme is based on Computational Security. However, with development of computing technologies, one must take care, when choosing the appropriate one-way function. Here, we present a solution that based on Information-Theoretic Security that remains safe even with respect to an adversary with unlimited computing power.

### Our Scheme with Information-Theoretic Security

We use the One-time-pad to make group access secret dispersion method satisfy the Information-Theoretic Security. However, there is a drawback of using One-time-pad, this drawback is the encryption key length required and the length of plaintext that need to be encrypted is the same civilization. This may result that the space we need to store the key storage space much larger than other normal encryption technology.

***Secret Sharing***

1. Dealer generate a random number $r$ that has the same length to $v$.
2. Dealer calculates $q$ follow the below formula.

$$q = v \oplus r$$

(Here, $\oplus$ is bitwise exclusive-OR operation and $r$ is random number that has the same length with $v$)

3. Dealer calculates disguised data $p$ follow the below formula.

$$p = s - q$$

4. Dealer make $p_i$ and $q_j$ by using $p$ and secreted authority secret $q$ in many share in Shamir secret sharing scheme.
5. Dealer distributes pairs $(p_i, q_j)$ to groups.

When someone wants to reconstruct the secret $s$, he should have the right to access $v$. In order to access $v$, he should ask Dealer to request $r$ to access $v$.

***Secret Reconstruction***

1. Recoverer gets pairs ($p_i$, $v_j$) from different groups.
2. Recoverer reconstruct $p$ by using $p_i$. (The number of $p_i$ must exceed $t$).
3. Recoverer reconstruct $q$ by using $q_j$. (The number of $q_j$ must exceed $k$).
4. Recoverer uses $r$ to recover authority secret.

$$v = q \oplus r$$

5. Finally, Recoverer get secret by using above formula.

$$s = p + v$$

## 5. SECURITY ANALYSIS FOR THE CROSS-GROUP SECRET SHARING SCHEME

In a Cross-group secret sharing scheme, in order to recover data secret correctly, the ($t$, $m$)-($k$, $n$)-threshold scheme which combines two Shamir secret sharing schemes, and the two threshold schemes must be satisfied at the same time.

Our Proposal's security is based on three factors.

1. The security of random $r$ or key $h$, *which* was hold by Dealer.
2. The security of data shares $p_i$ which were distributed to different groups' servers.
3. The security of authority shares $v_i$ which were distributed to different groups' servers.

**Definition 5.1** Let $m \geq 2$, $m$ is the number of group, $t \leq m$, $t$ is the authority secret. Let $n \geq m$, $n$ is the number of servers, every group has least of one server, and $k$ is the threshold of data secret. The set of authority shares is defined for $VV = \{ v_1, v_2, \ldots, v_m \}$ and The set of data shares is defined for $PP = \{ p_1, p_2, \ldots, p_n \}$

**Example 5.1**

Let us consider $t = 2$, $m = 3$, $k = 7$, $n = 10$ which we can see the model in Fig.1.

**Situation 1** The adversary has the random $r$ or key $h$. but has no ability to get authority shares $v_i$ and data share $p_i$.

It is obviously that the adversary cannot reconstruct data secret $s$, he only known the random $r$ or key $h$. Because of equation $s = p + q$, both $p$ and $q$ are unknown to the adversary, so he cannot reconstruct data secret $s$.

**Situation 2** The adversary has not the random $r$ or key $h$, but has the ability to get over $t$ authority shares $v_i$.

The adversary has authority shares $v_1$ and $v_2$. However, he cannot get data shares over threshold.

The adversary can use polynomial interpolation to reconstruct authority secret $v$.

$$VV = \{ v_1, v_2 \} \rightarrow v$$

However, the adversary cannot calculate $q$, because he has no ran dom $r$ or key $h$.

$$v \nrightarrow q$$

Because adversary has no more than 6 data shares, he cannot reconstruct data secret $p$.

$$PP = \{ p_1, p_2, \ldots, p_i \} \nrightarrow p \ \{ i \leq 6 \}$$

In the equation $s = p + q$, both $p$ and $q$ are unknown to adversary, he could not reconstruct data secret data $s$.

**Situation 3** The adversary has not the random $r$ or key $h$, but has the ability to get over $k$ data shares $p_i$.

The adversary has data shares $p_1, p_2, p_3, p_4, p_5, p_6, p_7$. However, he cannot get data shares over threshold.

The adversary can use polynomial interpolation to reconstruct data secret $p$.

$$PP = \{ p_1, p_2, p_3, p_4, p_5, p_6, p_7 \} \rightarrow p$$

Because adversary has no more than 2 authority shares, he cannot reconstruct authority secret $v$.

$$VV = \{ v_1, \ldots, v_i \} \nrightarrow v \ \{ i \leq 1 \}$$

Because the adversary has no random $r$ or key $h$ and authority secret $v$, he cannot calculate $q$.

In the equation $s = p + q$, $q$ is unknown to adversary, he couldn't reconstruct data secret data $s$.

**Situation 4** The adversary has no random $r$ or key $h$, but he has the ability to get over $t$ authority shares $v_i$ and data shares $p_i$.

The adversary has got data shares $p_1, p_2, p_3, p_4, p_5, p_6, p_7$, and has got authority shares $v_1$, $v_2$.

The adversary can use polynomial interpolation to reconstruct data secret $p$.

$$PP = \{ p_1, p_2, p_3, p_4, p_5, p_6, p_7 \} \rightarrow p$$

The adversary can use polynomial interpolation to reconstruct authority secret $v$.

$$VV = \{ v_1, v_2 \} \rightarrow v$$

Because the adversary has no random $r$ or key $h$ and authority secret $v$, he cannot calculate $q$.

In the equation $s = p + q$, $q$ is unknown to adversary, he could not reconstruct data secret data $s$.

**Situation 5** The adversary has random $r$ or key $h$, and he has the ability to get over $t$ authority shares $v_i$ and data shares $p_i$.

The adversary has data shares $p_1, p_2, p_3, p_4, p_5, p_6, p_7$, and has authority shares $v_1$, $v_2$.

The adversary can use polynomial interpolation to reconstruct data secret $p$.

$$PP = \{ p_1, p_2, p_3, p_4, p_5, p_6, p_7 \} \rightarrow p$$

The adversary can use polynomial interpolation to reconstruct authority secret $v$.

$$VV = \{ v_1, v_2 \} \rightarrow v$$

The adversary can calculate $q$, because of random $r$ or key $h$.

$$v \rightarrow q$$

In the equation $s = p + q$, both $p$ and $p$ are known to adversary, he could reconstruct data secret data $s$.

Seeing above 5 situations, only situation 5, and the adversary can reconstruct data secret $s$. If an adversary wants to break our scheme, he would obtain three factors that is a very hard thing to him. First, to get random $r$ or key $h$, he needs to corrupt Dealer to get it or stolen it when Dealer transfers it to third party. Usually, if the Dealer was corrupted, the all of scheme is done. Therefore, there is not mean to discuss how to protect the random $r$ or key $h$ when Dealer was corrupted. On the other hand an adversary may stole random $r$ or key $h$ from the process when Dealer transfers it to third party. However, Dealer transfers random $r$ or key $h$ to third party must encrypted it, and adversary should have a method to crack it. Second, to get data secret $p$, adversary should invade $t$ servers which $t$ servers are come from $k$ groups to reconstruct $p$. Third, adversary also should get $k$ groups' authority shares $v_i$ to reconstruct authority secret $v$. It is very difficult to satisfy three conditions at the same time, so we could conclude that our proposal have a high level of security in cross-group secret sharing scheme.

## 6. REVOKE LEAKED SHARES

A robust scheme must have ability to defend the attack comes from adversary. In some case, an adversary may be able to obtain some authority share $v_i$ and data share pi. Nevertheless, before adversary get enough authority share $v_i$ and data share $p_i$ to reconstruct authority secret $v$ and data secret $p$. The scheme should update authority share $v_i$ and data share $p_i$, but cannot change the value of authority secret $v$ and data secret $p$. When shares were updated correctly, even adversary get the updated shares, and the number of shares (old shares and update shares) over threshold, adversary could not reconstruct secret that means leaked share were revoked when shares were updated. How to update shares? A simple idea is Dealer redistribute shares to each groups' servers and all of old shares are deleted by Group's managers and no server can store old share any more when delete command was performed. This idea is based on Dealer is always exist in scheme. Nevertheless, in some case, once Dealer distributed shares, he may leave the scheme. In this case, we could not expect Dealer come back and redistribute shares again. To update shares, shareholders (different groups' servers) have to assemble enough shares (i.e., the number of shares must over threshold) to reconstruct data secret $s$ or reconstruct authority secret $v$ and redistribute new shares again. However, if we do that, it will have a risk of leak data secret $s$ or authority secret $v$ in the process of reconstruction. To update shares without reconstruct data secret $s$ is expected. In the paper [3], Herzberg et al. gave us a feasible proposal to solve this problem.

### Herzberg et al.'s Proactive Share Update Proposal

We assume a system of n participates $A = \{A_1, A_2, \dots, A_n\}$ that will (proactively) share a secret value $x$ though a $(k, n)$-threshold scheme. The shares is descripted to $x_i$.

1. Shareholder $A_i$ generates a $k - 1$ order polynomial which have $k - 1$ random Coefficient $b_{ij}$.

$$b_i(z) = b_{i1}z + b_{i2}z^2 + \dots + b_{ik}z^{k-1}$$
$$b_i(0) = 0$$

2. Shareholder $A_i$ distribute $u_{ij} = b_i(j)$ to other shareholder $P_j$.
3. Shareholder $P_j$ use all of $u_{ij}$ to renew share.

$$W_j^{(t)} = W_j^{(t-1)} + (u_{1j} + u_{2j} + \dots + u_{nj}).$$

If all of shareholders perform above procedures correctly. All of shares will be updated correctly. If some shareholders give some wrong renew information $u_{ij}$. The shares will not be updated correctly. To solve this problem, in the paper [4] Feldman gave us a proper method to check whether the update information is correct or not.

Using Herzberg et al.'s Proactive Share Update Proposal, we need not reconstruct authority secret $v$ or data secret $s$, then rebuild secret shares again and redistribute to shareholders. The merit of Herzberg et al.'s Proactive Share Update Proposal is adversary cannot get the authority secret $v$ or data secret $s$ in the reconstruction of authority secret $v$ or data secret $s$. However, as we see, with the number of participants increasing, Herzberg et al.'s Proactive Share Update Proposal will waste a large of time. Therefore, we can conclude that Herzberg et al.'s Proactive Share Update Proposal is not effective. Balance with the security and effective, the number of participants should be control in a proper value.

## 7. ADD NEW SHARE

As a commerce cloud data storage service, the service must have the ability of add new share to new server when a group is become large and Dealer wants to give more shares to the group. Let me think about that, when a provider is becoming a reliable people more than other providers, which we may allow the provider to have more shares than other providers are. However, once data secret shares and authority shares were distributed to providers, there is no method to add new shares without reconstruct data secret to provider in most previous work [6] [7] [8] [9]. That means when a provider purchased new servers and Dealer give him more right which means the provider can add new shares to new servers. If the provider wants to add a new share, he should get $t$ data shares and reconstruct data secret and rebuild $n + 1$ shares and redistribute to servers. There is a problem that if the provider was corrupted, adversary may cheat data secret or authority from the provider. To prevent it, we are expected a method that without reconstruct to data secret or authority secret, we can build a new share from old shares without Dealer's cooperation. In the paper [15], F.Bao gave us a possible method to solve this problem.

**Protocol**

1. For each $i$ ( $i = 1, \dots, n$ ), $A_i$ randomly chooses $b_{i1}, b_{i2}, \dots, b_{it}$ from $Z_p$, and set

$$f_i(x) = b_{i1}x + b_{i2}x^2 + \dots + b_{it}x^t - \sum_{j=1}^{t} b_{ij}y_r^j.$$

That is, $A_i$ randomly chooses $g_i(x)$ from $Z_p[x]$ such that

$$f_i(y_r) = 0.$$

2. For each $i$ ( $i = 1, \dots, n$ ), $A_i$ sets $s_{ij} = f_i(x_j)$ and gives $s_{ij}$ to $A_j$ for $j \in \{1, \dots, n\}$.
3. Upon receiving $s_{1j}, \dots, s_{nj}$, $A_j$ defines a new share.

$$\overline{s_i} \leftarrow s_j + ( s_{1j} + \dots + s_{nj} )$$

And sends $\overline{s_i}$ to $V_r$.

4. Upon receiving $\overline{s}_1, \dots, \overline{s}_n$, $V_r$ solves the linear equations

$$\overline{s_i} = z_0 + x_i z_1 + x_i^2 z_2 + \dots + x_i^t z_t \quad (i = 1, \dots, n)$$

5. For $z_0, z_1, \dots, z_t$. Finally, $V_r$ sets

For $z_0, z_1, \dots, z_t$. Finally, $V_r$ sets

$$t_r = z_0 + y_r z_1 + y_r^2 z_2 + \dots + y_r^t z_t$$

Using protocol, we can add new share without reconstruct data secret or authority secret, which can reduce risk of leak secret.

## 8. CONCLUSION

In this paper, we have proposed a new two-thresholds $(t, m)$-$(k, n)$ secret sharing scheme which permits the data owner to impose a condition that the reconstructing shares come from more than one storage providers. This can be useful, if the data owner prefers not to rely solely on one provider. Our proposal is easy to implement, and it incurs a little overhead compared to the ordinary Shamir scheme. Moreover, when storing data on the internet, we cannot only care about confidentiality, but also should care about the availability of data – if some servers were destroyed e.g. by natural disaster, with remaining information we should be able to recover the secret.

## 9. ACKNOWLEDGMENTS

This project was partially supported by JSPS and DST under the Japan-India Science Cooperative Program.
The second author is partially supported by a *kakenhi* Grant-in-Aid for Scientific Research (C) 15K00029 from Japan Society for the Promotion of Science.
The fourth author is partially supported by a *kakenhi* Grant-in-Aid for Scientific Research (C) 15K00186 from Japan Society for the Promotion of Science.
The fifth author is partially supported by a *kakenhi* Grant-in-Aid for Scientific Research (C) 15H02711 from Japan Society for the Promotion of Science.

## 10. REFERENCES

[1] Shamir, A. 1979. How to share a secret. In *Communications of the ACM*. 22.11 (Nov. 1979), 612-613.

[2] Blakley, G. R. and Meadows, C. 1985. Security of ramp schemes. In *Advances in Cryptology. Lecture Notes in Computer Science*. 196, 242-268

[3] Desmedt, Y., and Jajodia, S. 1997. Redistributing secret shares to new access structures and its applications. *Technical Report ISSE*. 148, TR-97-01, George Mason University.

[4] Feldman, P. 1987. A practical scheme for non-interactive verifiable secret sharing. In *Proceedings of the 28th Annual Symposium on Foundations of Computer Science*. SFCS '87. IEEE Computer Society, Washington, DC, USA, 427-438.

[5] Herzberg, A., Jarecki, S., Krawczyk, H., and Yung, M. 1995. Proactive secret sharing or: How to cope with perpetual leakage. *Advances in Cryptology—CRYPT0'95*. 963,(Jul. 2001). 339-352.

[6] Chor, B., Goldwasser, S., Micali, S. and Awerbuch, B. 1997. Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults. *Foundations of Computer Science, 1985., 26th Annual Symposium on*. IEEE, Portland, 383-395.

[7] Pedersen, T. P. 1992. Non-interactive and information-theoretic secure verifiable secret sharing. *Advances in Cryptology—CRYPTO'91*. 576, (May. 2001), 129-140

[8] He, J., and Dawson, E. 1995. Multisecret-sharing scheme based on one-way function. *Electronics Letters* 31,2 (Jan. 1995), 93-95.

[9] Martin, K. M., Pieprzyk, J., Safavi-Naini, R., and Wang, H. 1999. *Information Security and Privacy*. 1587, (Jul. 2001), 177-191.

[10] Blakley, G. R. 1979. Safeguarding cryptographic keys. In *Proceedings of the national computer conference*. New York, 313.

[11] Li, B. 2006. A strong ramp secret sharing scheme using matrix projection. In *Proceedings of the 2006 International Symposium on World of Wireless, Mobile and Multimedia Networks*. IEEE, Buffalo-Niagara Falls, NY, 652-656.

[12] Pang, L., Li, H., Yao, Y., and Wang, Y. 2008. A verifiable (t, n) multiple secret sharing scheme and its analyses. In *Electronic Commerce and Security*. 2008 International Symposium on. IEEE, 22-26.

[13] Tompa, M., and Woll, H. 1989. How to share a secret with cheaters. *Journal of Cryptology* 1,3, 133-138.

[14] Ostrovsky, R., and Yung, M. 1991. How to withstand mobile virus attacks. In *Proceedings of the tenth annual ACM symposium on Principles of distributed computing*. PODC '91. ACM, New York, NY, 51-59.

[15] Bao, F. Deng, R. H., Han, Y. and Jeng, A. B. 1997. Design and analyses of two basic protocols for use in TTP-based Key escrow. In *Proceedings of the Second Australasian Conference on Information Security and Privacy.* ACISP '97. Springer-Verlag, London, UK, 261-270.

[16] Beimel, A. 1996. Secure schemes for secret sharing and key distribution. *Diss. Technion-Israel Institute of technology, Faculty of computer science*. 27-31.

[17] Sarrna, K. S., Larnkuche, H. S., and Urnarnaheswari, S. 2013. A Review of Secret Sharing Schemes. *Research Journal of Information Technology.* 5, 67-72.

[18] Smith, G., Boreli, R., and Kaafar, M. A. 2013. A Layered Secret Sharing Scheme and its Application to OSN Groups. *Mobile and Ubiquitous Systems: Computing, Networking, and Services*. 131 (Sep. 2014), 487-499.

[19] Zhang, Z., Chee, Y. M., Ling, S., Liu, M., and Wang, H. 2012. Threshold changeable secret sharing schemes revisited. *Theoretical Computer Science*, 418 (Feb. 2012), 106-115.

[20] Martin, K. M., Pieprzyk, J., Safavi-Naini, R., and Wang, H. 1999. Changing thresholds in the absence of secure channels. In *Information Security and Privacy*, 1587 (July 2001), 177-191.

[21] Iftene, S., and Boureanu, I. C. 2005. Weighted threshold secret sharing based on the Chinese remainder theorem. In *Scientific Annals of Cuza University*, 15, 161-172.

[22] The World's First High-speed Secret Sharing Engine for OpenStack Swift. DOI=http://www.ntt.co.jp/news2015/1505e/150518a.htm