情報処理学会論文誌 Vol.54 No.9 2146–2155 (Sep. 2013)

記憶容量削減と計算量的安全性および復元の独立性を実現するクラウドに適した秘密分散法

高橋 慧[1,a)] 小林 士郎[1] 岩村 恵市[1]

受付日 2012年12月2日, 採録日 2013年6月14日

概要：秘密分散法の活用法として，クラウドコンピューティングへの応用が考えられる．しかしShamirの秘密分散法をクラウドシステムへそのまま適用すると各サーバが持つ分散情報のデータサイズを秘密情報より小さくすることができないため，大量の秘密情報の分散を行う際には，多くのサーバが大量のデータを保存する必要がある．また，サーバの記憶容量を削減するためにランプ型秘密分散法を適用すると，段階的な情報漏えいが生じることが知られている．そこで，本論文ではクラウドシステムを想定し，数多くの小さなサイズの秘密情報を分散させた場合に適した分散情報の記憶容量削減と計算量的な安全性を実現する新たな秘密分散法の提案を行う．

キーワード：秘密分散法，ビッグデータ，クラウドコンピューティング，記憶容量削減

Secret Sharing Scheme Suitable for Cloud Computing Achieve Reduce Storage Capacity, Computational Safety and Reconstruct Independence

Satoshi Takahashi[1,a)] Shiro Kobayashi[1] Keichi Iwamura[1]

Received: December 2, 2012, Accepted: June 14, 2013

Abstract: Recently, secret sharing scheme attracting attention to apply cloud computing system. However, when Shamir's secret sharing is applied to cloud system, each server which compose the cloud system should store many data because, it cannot make size of share smaller than size of secret. Moreover, the case when Ramp scheme is applied to cloud system for reducing size of share, it gradual possible to obtain the secret from less than "k" pieces. Then, in this paper we assume specific cloud system and propose new secret sharing scheme which achieve computational secure while reducing amount of share when distribute many small size secrets. Also, we compare the size of share with each conventional methods.

Keywords: secret sharing, big data, cloud computing, reducing storage capacity

1. はじめに

近年，クラウドコンピューティングの利用が注目されている．クラウドコンピューティングとはユーザの持つデータをクラウドと呼ばれるネットワーク上の複数のサーバにより構成される仮想の大容量ストレージに分散・保管し，そのデータをどこからでもネットワーク経由でユーザが必要に応じてアクセスすることを可能にする技術である [1], [2].

このようなクラウドシステムを構成する際に秘密分散法を適用することが検討されている．秘密分散法を用いたクラウドシステムは以下のような要件を満たしていることが望ましいと考えられる．

(1) 1つのデータを複数のサーバに分散させ，これらのデータのうちいくつかが破損しても元のデータを復元することができる．

(2) 秘密分散法によって増加する記憶容量をできるだけ小

[1] 東京理科大学
Tokyo University of Science, Katsushika, Tokyo 125–8585, Japan
[a)] takahashi@sec.ee.kagu.tus.ac.jp

さくすることができる．

(3) 分散情報をシステムが定めた閾値以上集めることで，完全に秘密情報を復元でき，その閾値未満の分散情報が集まった場合でも秘密情報に関する情報漏えいが生じず，少なくとも計算量的安全性を持つ．

(4) ある秘密情報の復元を行った際にこの秘密情報の復元過程で他の秘密情報に関する情報を得ることができない．

まず (1) の要件について考える．クラウドコンピューティングは複数のサーバにより構成されており，これらのサーバのうちのいくつかが災害による停電やヒューマンエラーなどによりダウンすることは十分に考えられるため，この要件は必要である．次に，(2) の要件について考える．準備できるストレージサイズが，保存すべきデータ容量より十分大きければ，この要件は必要ではない．しかし，現在多くのクラウドシステムでは多数のユーザからの大量のデータ保存が想定されるため，大規模ストレージを持つサーバを用いてクラウドシステムが構成されており，クラウドの構成に大きなコストを必要とする．また，ユーザやデータ数は今後も増加し続けることが予想されるため，増加する記憶容量はできるだけ小さく抑えられることが望まれる．また，クラウドシステムを構成する主体として，参加ユーザのモバイル端末などを利用することを考える．たとえば，分散情報の一部をモバイル端末に持たせておき，必要に応じて残りの情報を他のサーバから取り寄せる場合などである．この場合，モバイル端末には大きな記憶容量を持たせることはできない．よって，モバイル端末のような特定の主体の記憶容量を小さく抑えることができることができれば，より柔軟なシステムを構築できるようになる．上記のように広い応用を考えれば，指定した特定のサーバの記憶容量が秘密分散により増加しないことも望まれるが，ここでは一般的に (2) の要件を持つことが望ましいとする．また，クラウドシステムはユーザに対して，預かったデータの安全性を保証する必要がある．一般に，金融システムなどのデータ管理においては公開鍵暗号などを利用した暗号化が施され，計算量的安全性が保証される場合が多い．よって，顧客からデータを預かるクラウドシステムにおいても少なくとも計算量的安全性が保証されている必要があると考える．できれば情報量的安全性を持つことが理想であるが，秘密分散法の種類によっては，データの小容量化により分散情報の一部が漏洩すると段階的に秘密情報が漏洩する手法がある．この場合，ある閾値までは秘密情報は完全に漏洩しないが，その閾値を超えると部分的であっても情報漏洩の可能性が生じる．このような性質は，ユーザのデータを預かるクラウドシステムとしては望ましくない．よって，(3) の要件を持つことが望ましいと考えられる．最後に (4) の要件については，クラウドコンピューティングでは複数のデータを分散させたユーザが自身のデータの一部を自身以外のユーザに公開することが考えられる．たとえば，経理担当者が入力した全社員の給与情報などについて，各社員が自身に関する情報のみ閲覧する場合などである．このような場合に，あるユーザ X のあるデータ d が別のユーザ Y に復元されたとき，Y が X のデータ d 以外に関する情報を得ることがあってはならない．このため，この要件はクラウドコンピューティングにおいて重要であり，本論文ではこの (4) の要件を「復元の独立性」と呼ぶ．

ここで，これまでに提案された代表的な秘密分散法をクラウドに適用することを考える．まず，(k, n) 閾値秘密分散法 [3] をクラウドに用いると，n 個のサーバにデータを分散させユーザはそのうち k 個を集めることで元の秘密情報を復元することができるため (1) の要件を満たすことができる．また，(3)，(4) の要件についてもこの手法は情報量的安全性を持つことが知られているため，これも満たすことができる．しかし，この手法では各サーバの持つ分散情報のデータサイズは秘密情報よりも大きいため，(2) の要件は満たしていない．そこで，この欠点を補う手法として，ランプ型秘密分散法 [4] を用いる場合，ユーザの記憶容量を (k, n) 閾値秘密分散法に比べ，$1/L$ 倍にすることができる．また，それぞれの秘密情報について独立に分散式を定めることで，復元を行うユーザは各秘密情報を独立に復元することができるが，$k - L + 1$ 個以上の分散情報からは秘密情報に関して段階的な漏洩が生じる．これより，この手法は (1)，(2) および (4) の要件は満たしているが，(3) の要件を満たしていない．また，(1)〜(3) の要件を満たすため，Krawczyk による方式 [6]（以降 Kr93 の方式）や千田らによる方式 [7]（以降千田方式）が提案されているが，これらの方式では，前述の第三者ユーザへの公開という部分については具体的に想定されておらず，(4) の要件を満たす具体的な根拠は示されていない．

そこで，本論文では上記の (1)〜(4) の要件を満たすことができる新しい秘密分散法の提案を行う．本提案方式では従来方式のように分散情報のデータサイズ自体を縮小するというアプローチではなく，サーバの持つ分散情報の個数を削減するというアプローチにより多くの小さなサイズの秘密情報を分散させた場合に，システム全体で必要となる記憶容量の削減を実現する．これによって，前述の特定のサーバの記憶容量が秘密分散によって増加しないという特徴も実現する．また，あるユーザが分散を行った秘密情報について攻撃者が $k - 1$ 台のサーバと結託した場合でもこれらの秘密情報は計算量的安全性を持つということを示す．

以下，本論文の構成を示す．2 章において本論文で想定するシステムについて説明を行う．その後 3 章では秘密分散法の従来方式について説明を行い，これらの方式を想定システムに適用した場合を考え，前述のクラウドに必要な

要件について考察を行う．4 章では前述の 4 つの要件すべてを満たすことができるクラウドへの適用に適した提案方式の概要について説明を行い，5 章ではその評価を行う．

2. 想定システム

図 1 および図 2 には本論文で想定するシステムにおける分散および復元時のシステム構成図を示す．

本想定システムでは，図 1 のように多くのユーザ（本論文では r 人とする）が参加する大規模なクラウドシステムを想定し，それぞれのユーザが多くのデータ（本論文では m 個とする）を n 台のサーバによりネットワーク上で構成されたクラウドシステムへ分散・保管する．

また，本想定システムでは図 2 のように秘密情報を復元するユーザはこれらのデータの分散を行った持ち主のみでなく，ユーザは第三者へ自身のデータの一部を公開することを想定する．

3. 従来方式

本章では，秘密分散法の基本的な手法である Shamir の (k,n) 閾値秘密分散法および (k,L,n) ランプ型秘密分散法の紹介を行い，5 章で比較を行う従来方式として秘密分散法と暗号技術を組み合わせた Kr93 の方式および秘匿計算への応用を想定した千田方式の説明を行う．また，これらそれぞれの方式について前述のクラウドに必要な要件について評価を行う．

なお，本論文全体を通じて秘密分散に関する計算は秘密情報を s，ユーザが秘密情報を分散するサーバの台数を n としたとき，$s<p$ かつ $n<p$ である素数 p による (mod p) 上で行うものとする．つまり本論文においてすべての秘密分散に関する計算は p 個の要素を持つ GF(p) 上で計算される．また，分散する秘密情報の個数を m とし，簡単のため全秘密情報と分散情報のサイズを同じとし，$|s|$ と表す．

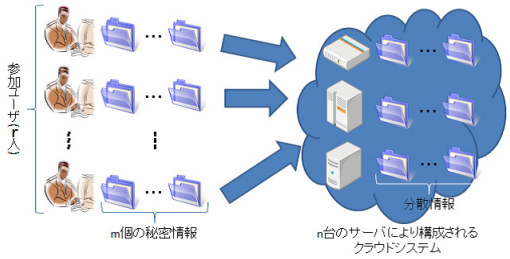

図 1 想定システムの分散時におけるシステム構成図

Fig. 1 Processing diagram of the assumed system in distribution phase.

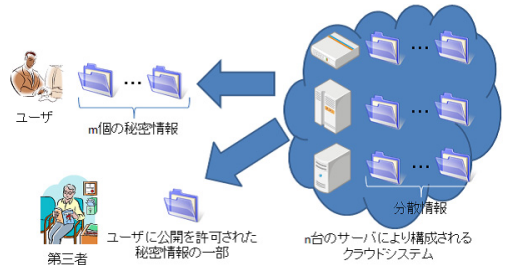

図 2 想定システムの復元時のシステム構成図

Fig. 2 Processing diagram of the assumed system in reconstruction phase.

3.1 (k,n) 閾値秘密分散法

次の 2 つの条件を満たす秘密分散法を，(k,n) 閾値秘密分散法と呼ぶ．

(1) k 個以上の任意の分散情報から，元の秘密情報 s を完全に復元することができる．

(2) $k-1$ 個以下の分散情報からは，秘密情報 s に関する情報はいっさい得ることができない．

Shamir の提案した多項式補間による方法では，以下のようにして (k,n) 閾値秘密分散法を実現する．

[分散]

(1) $s<p$ かつ $n<p$ である任意の素数 p を選ぶ．

(2) $GF(p)$ の元から，異なる n 個の x_i（$i=1,\cdots,n$）を選びだし，各サーバの識別子とする．

(3) $GF(p)$ の元から，$k-1$ 個の乱数 a_l（$l=1,2,\cdots,k-1$）を選んで，以下の式を生成する．

$$f(x)=s+a_1x+a_2x^2+\cdots+a_{k-1}x^{k-1} \tag{1}$$

(4) 上記式 (1) の x に各サーバの識別子 x_i を代入して，n 個の分散情報 $f(x_i)=W_i$（$i=1,2,\cdots,n$）を計算し，各サーバに x_i と生成した W_i を送信する．

[復元]

(1) 復元に用いる分散情報を n 個の分散情報 W_i（$i=1,2,\cdots,n$）から k 個選択し W_{if}（$f=1,2,\cdots,k$）とする．また，その分散情報に対応するサーバ識別子を x_{if} とする．

(2) 分散式 $f(x)$ に $x=x_{if}$ および $f(x_j)=W_{if}$ を代入し，k 個の連立方程式を解いて，秘密情報 s を得る．s の復元の際には，Lagrange の補間公式を用いると便利である．

この (k,n) 閾値秘密分散法を m 個の秘密情報に対して独立に行うことを考える．この場合，システム全体で必要な記憶容量は $m\times n\times|s|$ となる．

この方式は閾値 k 台以上のサーバの持つ分散情報を集めることで，元の秘密情報を完全に復元することができ，各分散情報についても情報量的安全性を持つことが知られている．また，この方式ではそれぞれの秘密情報ごとに独立に分散式が割り当てられているため，復元の独立性を保つことができる．しかしこの方式の場合，分散情報のデータサイズを元の秘密情報のデータサイズよりも小さくすることができないため，非常に容量効率が悪い．これより (k,n) 閾値秘密分散法は先ほどのクラウドとして満たしていることが望ましい要件の (1)，(3)，(4) を満たすことができ

るが，(2) については満たすことができないといえる．

3.2 (k, L, n) ランプ型秘密分散法

ランプ型秘密分散法は，以下の条件を満たす．

(1) 任意の k 個以上の分散情報から，元の秘密情報 s を完全に復元することができる．

(2) 任意の $k-t$ 個（$1 \leq t \leq L-1$）の分散情報からは，段階的に秘密情報 s の情報が得られる．

(3) $k-L$ 個以下の分散情報からは，秘密情報 s に関する情報はいっさい得ることができない．

多項式補間を利用した (k, n) 閾値秘密分散法を変更することで，以下のようにランプ型秘密分散法を実現できる．ここで秘密情報を $s=(s_0, s_1, \cdots, s_{L-1})$ とし，分散式を以下のように定めて，分散情報 W_i を計算する．

$$W_i = s_0 + s_1 x_i + s_2 x_i^2 + \cdots + s_{L-1} x_i^{L-1} + a_L x_i^L + \cdots + a_{k-1} x_i^{k-1} \quad (2)$$

復号の際には，(k, n) 閾値秘密分散法と同様の手順で連立方程式を解き $s_0, s_1, \cdots, s_{L-1}$ を求める．

この方式を用いた場合，m 個の秘密情報を分散した場合でもシステム全体で必要となる記憶容量は $(m \times n \times |s|)/L$ となるため，(k, n) 閾値秘密分散法を独立に繰り返した場合に比べ，システム全体の記憶容量を $1/L$ に削減できる．また，各秘密情報について独立に多項式を定めることで，復元の独立性を保つことができる．しかし，この方式は復元過程において $k-L+1$ 個以上の分散情報が集まると段階的に秘密情報が漏えいすることが知られている [7]．これより，ランプ型秘密分散法は前述の要件 (1)，(2)，(4) を満たすことができるが，(3) の要件については満たすことができないといえる．

3.3 Kr93 の方式

Kr93 の方式は，秘密情報を暗号化し，暗号化に必要な暗号鍵を (k, n) 閾値秘密分散法を用いて保管すると同時に，暗号文をランプ型秘密分散法を用いて保管する方式である．この手法では秘密情報の暗号化に計算量的安全性を持つ暗号装置を利用し，暗号鍵が安全に管理されている場合には，たとえ前述のランプ型秘密分散法により分散されている分散情報が閾値未満集まった場合でも，暗号化された暗号文に関する情報が漏洩するのみであり，少なくとも秘密情報について計算量的安全性を保つことができる．

この手法において m 個の秘密情報を 1 つの鍵を用いて暗号化する場合を考える．ここで，鍵サイズを $|key|$ とすると，システム全体で必要となる記憶容量は $(n \times m \times |s|)/L + n \times |key|$ となる．よって，この手法は m が十分大きく，鍵サイズを無視できるとすると，高い容量削減効果を持つ．

しかし，この方式においてこのように複数の秘密情報を 1 つの暗号鍵ですべての秘密情報を暗号化した場合，1 度秘密情報を復元したユーザは他の秘密情報に関する情報を知ることになるため，このユーザにとって他の秘密情報の安全性がランプ型秘密分散法と同等となり，安全性に問題が生じる．このため，Kr93 の方式では「復元の独立性」を実現することができない．

これより，Kr93 の方式は複数の秘密情報を 1 つの暗号鍵で暗号化する場合には前述の要件の (1)，(2)，(3) を満たすことができるが，(4) の要件は満たさないということになる．また，秘密情報それぞれを独立の暗号鍵を用いて暗号文を生成し，分散を行った場合，秘密情報のデータサイズが利用する暗号鍵のデータサイズよりも十分大きい場合は暗号鍵の分散に必要な分散情報の大きさを無視することができるため，(1)〜(4) の要件を満たすことができるが，秘密情報のデータサイズと暗号鍵のデータサイズに大きな差がない場合には鍵の分散に必要な分散情報のデータサイズを無視することができないため，(2) の要件を満たすことができない場合がある．

3.4 千田方式

千田方式は元の秘密情報から n 個の分散情報を生成する分散処理，k 個の分散データから秘密情報を復元する復元処理，そして n 個の分散情報を shamir 秘密分散の分散情報に変換する変換処理からなり，以下のように構成される．

[分散処理]

(1) 環 S 上の元データ $s \in S$ を入力する．

(2) $k-1$ 個のシード $r_1, \cdots, r_{k-1} \in R$ および擬似乱数生成関数 $P : R \to S$ を用いて $f_s(i) = P(r_i)$ $(i = 1, \cdots, K-1)$ を計算する．

(3) $f_s(K) = s - \sum_{i=1}^{k-1} f_s(i)$ を計算する．

(4) S 上の任意の (k, n) 閾値秘密分散法を用いて $r_1, \cdots, r_{K-1}$ を分散し，$f_s(k)$ を IDA [9] を用いて分散し，これらの分散情報の組みを s の分散情報とする．

[復元処理]

(1) s の k 個の異なる分散情報から $r_1, \cdots, r_{K-1}$ および $f_s(k)$ を復元する．

(2) $f_s(i) = P(r_i)$ $(i = 1, \cdots, K-1)$ を計算する．

(3) $s = \sum_{i=1}^{k} f_s(i)$ を復元する．

ここで，千田方式を複数の秘密情報の分散に適用することを考える．千田方式の場合，擬似乱数のためのシード $r_1, \cdots, r_{k-1}$ をすべての秘密情報に共通に用いているため，1 度これらのシードを復元したユーザに対して他の秘密情報に関する情報が漏えいすることになり，安全性が低下する．このため，千田方式で「復元の独立性」を実現するためにはこれらのシードは各秘密情報ごとに独立に設定される必要があり，サーバは多くの情報を持つ必要がある．このため，千田方式も前述の Kr93 の方式と同様すべての秘密情報に対して 1 つのシードのみを割り当てる場合には要

件の (1), (2), (3) を満たすことができるが, (4) の要件は満たさないということになる. また, m 個の秘密情報それぞれに独立にシードを割り当てる場合には (1), (3), (4) の要件を満たすことができるが, Kr93 の方式と同様, 千田方式もシードの分散の際に (k,n) 閾値秘密分散法を用いるため, (2) の要件については分散を行う秘密情報の個数やデータサイズなどに依存する形となる.

4. 提案方式

前述の各従来方式では各サーバの持つデータ量削減のため, 各サーバの持つ分散情報自体のデータサイズを縮小することにより記憶容量の削減を行った. これに対して本提案方式ではサーバの持つ分散情報の個数を削減することにより, システム全体でサーバが持つべきデータ量の削減を実現する.

4.1 提案方式の概要

本提案方式では**図 3** に示すように図 1 の n 台のサーバから l 台を選択し, 鍵サーバとする. これらの鍵サーバは分散情報を持たず, 擬似乱数を生成するための鍵情報のみを持つため, 分散情報の個数が削減される. 鍵サーバはユーザの要求に応じて擬似乱数を生成し, ユーザはその擬似乱数を分散情報として分散式を決定する. そして, この際決定された分散式を用いて残りのサーバの持つ分散情報の算出を行う. ここで, このような鍵サーバ以外のすべての秘密情報に関する分散情報を保管するサーバをデータサーバと呼ぶ. また, データを分散する各ユーザにはユーザ識別のため $ID[y]$ $(y=1,\cdots,r)$ が割り当てられており, それぞれのユーザが持つ m 個の秘密情報 $s_{1j}\cdots s_{mj}$ $(j=1\cdots r)$ にもそれぞれデータ識別のため $dID[s_{ij}]$ $(i=1,\cdots,m)$ が割り振られているものとする. このような ID は図 1 のような複数のユーザが複数のデータを分散するシステムでは, どのよな秘密分散法を用いた場合でも必要となる. 従来方式ではそのようなクラウドシステムへの適応が想定されておらず, 1 人のユーザが 1 つのデータを分散させることを前提として説明されているので, このような ID は想定されていない. 提案方式ではこれらの ID 情報を用いて擬似乱数を生成することで, それぞれの秘密情報を独立に復元することが可能となっている.

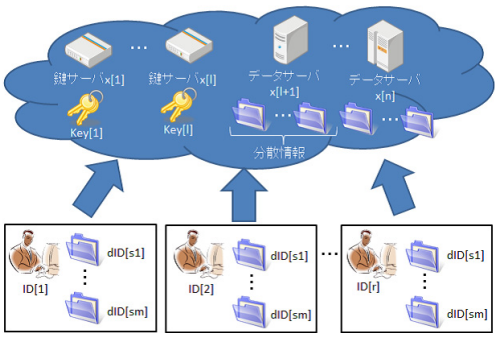

図 3 提案方式におけるシステム構成図
Fig. 3 Processing diagram of proposed method.

4.2 提案方式の構成

まず, それぞれの秘密情報 s_{ij} $(i=1,\cdots,m,\ j=1,\cdots,m)$ について以下の分散式 $f(x)$ を生成する. ここで, 提案方式においてもすべての計算は GF(p) 上で行うものとする.

$$f(x)=s_{ij}+a_{i1}x+a_{i2}x^2+\cdots+a_{ik-1}x^{k-1} \tag{3}$$

各サーバには上記の分散式 $f(x)$ にそれぞれのサーバ識別子 x_j を代入したときに得られる値 $f(x_j)=W_{ij}$ を分散情報 W_{ij} として送信する. また, それぞれの秘密情報に割り振られているデータ識別子 $dID[s_{ij}]$ は秘密情報のデータサイズよりも小さいものとし, これらの $dID[s_{ij}]$ および秘密情報 s_{ij} の間には以下の関係があるとする. ただし, $H(A)$ は A という情報に関するエントロピーを表し, $H(A|B)$ は B という情報を知ったときの A に関するエントロピーを表す.

$$H(s_{ij}|dID[s_{ij}])=H(s_{ij}) \tag{4}$$

本提案方式はユーザが鍵情報のみを持つ鍵サーバにそれぞれのユーザの識別のために割り振られたユーザ識別子 $ID[y]$ $(y=1,\cdots,r)$ を送信し, それを受け取った鍵サーバが自身の持つ鍵 key_j と受け取った $ID[y]$ を用いて $Eid(y,j)=Enc(ID[y],key_j)$ $(j=1\cdots l)$ を生成してユーザに送信する. ここで $Enc(a,b)$ は a を b という鍵を用いて暗号化する処理を表すとする. ユーザがこれを用いて自身のデータ識別子 $dID[s_{ij}]$ を暗号化し暗号化結果

$$q_{ij}=Enc(dID[s_{ij}],Eid(y,j))$$

を m 個の秘密情報すべてについて生成したのち, これらが鍵サーバの分散情報に対応するように $W_{1j}=q_{1j}$, $W_{2j}=q_{2j},\cdots,W_{mj}=q_{mj}$ としてそれぞれの秘密情報 s_i $(i=1\cdots m)$ に関する分散式 (3) 中の係数 $a_{i1},\cdots,a_{ik-1}$ を定める. 以上を一般的に書くと提案方式の構成は以下のようになる.

ここで, 分散情報を削減する鍵サーバの台数はクラウドシステム構成時に決定しており, l 台 $(2\le l\le k)$ とする. また, それぞれの秘密情報における分散式中の各係数を k 次のベクトルを用いて $A(i)=[s_{ij},a_{i1},\cdots,a_{ik-1}]^T$ と表す.

[分散]

(1) ユーザは自身の $ID[y]$ $(y=1,\cdots,r)$ を鍵サーバ $x_1,\cdots,x_l$ に送信する.

(2) $ID[y]$ を受け取った鍵サーバは自身の持つ暗号装置と鍵 key_j を利用して $Eid(y,j)=Enc(ID[y],key_j)$ $(j=1,\cdots,l)$ を生成し, ユーザに送信する.

(3) これを受け取ったユーザは自身の秘密情報に関するデータ識別子 $dID[s_{ij}]$ $(i = 1 \cdots m)$ を用いて擬似乱数

$$q_{ij} = Enc(dID[s_{ij}], Eid(y, j))$$

を生成する．

(4) ユーザはまず前述の k 次の分散式の係数ベクトル $A(i) = [s_{ij}, a_{i1}, \cdots, a_{ik-1}]^T$ における $k-1-l$ 次の部分ベクトル $A'_{k-1-l}(i) = [a_{il+1}, \cdots, a_{ik-1}]^T$ を真性乱数を用いて定める．その後，手順 (3) で生成した擬似乱数系列

$Q = [q_{1j}, \cdots, q_{mj}]^T$ および鍵サーバの ID 系列

$$X' = \begin{bmatrix} x_1 & \cdots & x_1^{k-1} \\ \vdots & \ddots & \vdots \\ x_l & \cdots & x_l^{k-1} \end{bmatrix} \quad (5)$$

を用いて以下の式から分散式の係数ベクトル $A(i)$ における残りの部分ベクトル $A'_l(i) = [a_{i1}, \cdots, a_{il}]^T$ を算出する．

$$A'_l(i) = X'^{-1}Q \quad (6)$$

これにより，ユーザは k 次の分散式の係数ベクトル $A(i) = [s_{ij}, a_{i1}, \cdots, a_{ik-1}]^T$ における $k-1$ 次の部分ベクトル $A(i)_{k-1} = [a_{i1}, \cdots, a_{ik-1}]^T$ を決定することができる．

(5) また，ユーザはデータサーバ $x_{l+1}, \cdots, x_n$ に関する分散情報 $W_{il+1}, \cdots, W_{in}$ を手順 (4) で生成した係数行列を利用して (k, n) 閾値秘密分散法と同様の手順により算出する．

(6) ユーザはそれぞれのデータサーバに生成した分散情報 $W_{1j}, \cdots, W_{mj}$ $(j = l+1, \cdots, n)$ を送信する．

[復元]

(1) 秘密情報 s_i を復元するユーザは n 個のサーバ $x_1, \cdots, x_n$ から任意の k 個のサーバを選択し，選択したサーバに対して自身の $ID[y]$ および秘密情報 s_i のデータ識別子 $dID[s_{ij}]$ を送信する．

(2) 鍵サーバの中で，$(ID[y], dID[s_{ij}])$ を受け取ったサーバは自身の持つ鍵 key_j および暗号装置を利用して

$$Eid(y, j) = Enc(ID[y], key_j)$$

を生成し，擬似乱数

$$q_{ij} = Enc(dID[s_{ij}], Eid(y, j))$$

を生成してユーザに送信する．

(3) データサーバの中で，$(ID[y], dID[s_{ij}])$ を受け取ったサーバはこれらの ID 情報に対応する分散情報 W_{ij} をユーザに送信する．

(4) サーバにより生成された分散情報および擬似乱数を受け取ったユーザは，これらを利用して (k, n) 閾値秘密分散法と同様の手段で，秘密情報 s_i を復元する．

これより提案方式では各ユーザの $ID[y]$ および秘密情報 s_{ij} に割り振られた $dID[s_{ij}]$ を利用することで，鍵サーバが持つ情報は擬似乱数生成のための key_j のみでよく，この鍵情報を鍵サーバが安全に保管するという前提をおいた場合，すべてのユーザに共通で利用することができるため，ユーザの人数および秘密情報の数に依存せずつねにただ 1 つの鍵情報のみを持つだけでよいということになる．

5. 評価

5.1 記憶容量の評価

本節では提案方式を用いてクラウドシステムを構成する場合に，システム全体でサーバが持つべき記憶容量の評価を行う．その後，各種従来方式を用いた場合にシステム全体で必要となる記憶容量と比較を行う．

提案方式を用いた場合，擬似乱数生成のための鍵情報のみを持つ鍵サーバとすべての秘密情報に関する分散情報を持つデータサーバの 2 通りのサーバが考えられる．まず，各鍵サーバの保管するデータは鍵情報 key_j のみとなる．ここで，比較の簡単化のため鍵情報のデータサイズは実数 t を用いて，$|key_j| = |s|/t$ であるとする．次にデータサーバは $m \times |s|$ の記憶容量が必要となる．これに加え，それぞれのデータサーバは各データ識別のための ID 情報を保管する必要があるが，これらの ID 情報はユーザが複数参加するシステムにおいてはどのような手法を用いた場合でも共通に保管する必要がある情報であるため，ここでは必要な記憶容量として含めずに考えるものとする．これより，提案方式において鍵サーバを l $(2 \leq l \leq k-1)$ 台として提案方式を利用して構成されたシステム全体で必要な記憶容量は以下のようになる．

$$\frac{|s|}{t} \times l + (n - l) \times (m \times |s|) \quad (7)$$

ここで，提案方式を用いた場合，鍵サーバの記憶容量はユーザの参加人数に依存せずつねに鍵情報 1 つ分のみ，つまり $|key_j| = |s|/t$ となる．これより，クラウドシステムに参加するユーザの人数を r 人とすると提案方式を利用した場合にシステム全体で必要な記憶容量は以下のようになる．

$$\frac{|s|}{t} \times l + r \times (n - l) \times m \times |s| \quad (8)$$

5.1.1 各種従来方式との記憶容量の比較

本項では前述の従来方式を代表して，Kr93 の方式，千田方式それぞれを用いて想定システムを構成した場合と，提案方式を用いて想定システムを構成した場合にシステム全体で必要な記憶容量の比較を行う．

ここで，今回想定するクラウドシステムでは，あるユーザがクラウドに預けたデータを他のユーザに公開するという場合を想定している．しかし，Kr93 の方式および千田方式ではそのような場合を想定しておらず，復元の独立性

を実現するための構成が示されていない．よって，まず従来方式で想定された複数の秘密情報すべてについて共通の暗号鍵を割り当てた場合，すなわち復元の独立性を実現しない場合について評価し，その次に復元の独立性を実現するために，その単純な拡張であるそれぞれの秘密情報について独立に暗号鍵やシードを割り当てる場合について評価を行う．また，今回の比較では，Kr93 の方式において利用するランプ型秘密分散法における L の値はそのとりうる最大値である $L = k$ であるとし，提案方式における鍵サーバの台数についても，そのとりうる最大値である $l = k - 1$ であるとして比較を行う．

そこでまず，Kr93 の方式について考える．この方式において 1 人のユーザが分散させる m 個の秘密情報について共通の鍵情報を利用した場合にシステム全体で必要な記憶容量は以下のようになる．

$$|s|/k \times n \times m \times r + |s|/t \times n \times r \tag{9}$$

このとき，Kr93 の方式と提案方式の記憶容量の比較をすると前述の式 (8) より以下のようになる．ここで，以降記憶容量の比較は比較対象となる従来方式を分母，提案方式を分子として行うものとする．

$$\frac{\frac{l}{t} + m \times r(n-k+1)}{(\frac{m}{k} + \frac{1}{t}) \times n \times r} \tag{10}$$

ここで，具体的な値を用いて記憶容量の比較を行う．想定する状況は，たとえば，企業における経理担当などが従業員の給与明細情報をクラウドへ分散させ，各従業員が自身の明細を閲覧する場合，または医者が複数の患者の血圧やヘモグロビン値などの診断値をクラウドに分散させて，各患者が自身の診断値を閲覧する場合や，ある会員が自身の属性情報や趣味などの個人情報をクラウドへ分散させ，特定の会員に閲覧を許可するなど，各種情報を個別に分散させる場合を考える．すべてのデータをまとめて 1 つの大きなデータとすることも考えられるが，これでは閲覧データを細かく制御することができない．よって，秘密情報は比較的データサーズの小さな情報であり，そのような情報を複数分散させることを想定する．このとき，秘密情報の個数 $m = 1000$，クラウドに参加するユーザ数 $r = 10000$ とし，鍵サイズについては秘密情報が比較的小さなデータサイズを想定しているため，$t = 2$ 程度つまり秘密情報の 1/2 の大きさであると仮定する．このとき，記憶容量の比較を行った結果を**表 1** に示す．

次に Kr93 の方式において復元の独立性を保つために，それぞれの秘密情報について独立に鍵を割りたてた場合を考える．この場合，システム全体で必要な記憶容量は以下のようになる．

$$|s|/k \times n \times m \times r + |s|/t \times n \times m \times r \tag{11}$$

これより提案方式に対して，K93 の方式においてそれぞれの秘密情報に対して独立に鍵を割り当てた場合の記憶容量の比較を行うと前述の式 (8) より以下のようになる．

$$\frac{\frac{l}{t} + m \times r(n-k+1)}{(\frac{1}{k} + \frac{1}{t}) \times n \times m \times r} \tag{12}$$

ここで，この場合も同様に前述と同じ条件においてそれぞれの n および k に対する具体的な容量の比較を行った結果を**表 2** に示す．

この結果より，Kr93 の方式は複数の秘密情報に対して共通に鍵を割り当てた場合には提案方式よりも良い容量削減効果を持つが，今回想定しているように利用する鍵のデータサイズに対して分散させる秘密情報のデータサイズが十分大きくない場合に，それぞれの秘密情報について復元の独立性を保つため，鍵を独立に定めた場合には提案方式は Kr93 よりも良い容量削減効果を持つことが分かる．また，鍵を独立に定めた場合には，サーバの台数 n が小さいほど提案方式の方が Kr93 の方式よりも容量削減効果が大きいということが分かる．ここで，具体的なクラウドシステムを考えた場合，これらのサーバの設置場所の確保，設備設置費用，メンテナンス費などを考えると，$n > 10$ というのは現実的ではないと考えられる [10]．これより，提案方式は十分な容量削効果を持つということができる．

次に千田方式について考える．千田方式についても前述の Kr93 と同様シードをすべての秘密情報に対して共通に割り当てた場合とそれぞれの秘密情報に対して独立に割り当てた場合について比較を行う．

まず，千田方式においてそれぞれの秘密情報に対して共通のシードを割り当てた場合にシステム全体で必要な記憶

表 1 提案方式と Kr93 の方式の記憶容量の比較（共通の鍵を用いた場合）

Table 1 Comparison of share size of proposed method and Kr93 (In case which using common key for all secrets).

$n = 6$ のとき		$n = 10$ のとき		$n = 20$ のとき		$n = 50$ のとき	
k	容量比	k	容量比	k	容量比	k	容量比
2	166%	2	180%	2	190%	2	196%
3	200%	4	279%	5	399%	5	459%
4	200%	6	299%	10	547%	10	816%
5	166%	8	239%	15	447%	20	1228%

表 2 提案方式と Kr93 の方式の記憶容量の比較（独立に鍵を割り当てた場合）

Table 2 Comparison of share size of proposed method and Kr93 (In case which using different key for each secret).

$n = 6$ のとき		$n = 10$ のとき		$n = 20$ のとき		$n = 50$ のとき	
k	容量比	k	容量比	k	容量比	k	容量比
2	83%	2	90%	2	95%	2	98%
3	80%	4	93%	5	114%	5	134%
4	67%	6	85%	10	91%	10	137%
5	48%	8	48%	15	53%	20	113%

表 3 提案方式と千田方式の記憶容量の比較（共通の鍵を用いた場合）
Table 3 Comparison of share size of proposed method and chida's method (In case which using common key for all secrets).

$n=6$ のとき		$n=10$ のとき		$n=20$ のとき		$n=50$ のとき	
k	容量比	k	容量比	k	容量比	k	容量比
2	166%	2	180%	2	190%	2	196%
3	199%	4	277%	5	392%	5	451%
4	198%	6	294%	10	505%	10	752%
5	163%	8	227%	15	372%	20	899%

表 4 提案方式と千田方式の記憶容量の比較（独立に鍵を割り当てた場合）
Table 4 Comparison of the share size of proposed method and chida's method (In case which using different key for each secret).

$n=6$ のとき		$n=10$ のとき		$n=20$ のとき		$n=50$ のとき	
k	容量比	k	容量比	k	容量比	k	容量比
2	56%	2	60%	2	63%	2	65%
3	29%	4	22%	5	19%	5	22%
4	15%	6	14%	10	6%	10	9%
5	8%	8	4%	15	2%	20	3%

容量を以下に示す．

$$|s|/k \times n \times r \times m + |s| \times (k-1) \times n \times r \tag{13}$$

これよりこの場合に千田方式において必要な記憶容量と提案方式において必要な記憶容量の比較を行うと前述の式 (8) より以下のようになる．

$$\frac{\frac{l}{t} + r \times m \times (n-k+1)}{n \times r \times m \times (\frac{1}{k} + k - 1) + \frac{n}{t}} \tag{14}$$

ここで，前述と同じ条件でそれぞれの n および k に対する具体的な容量の比較を行った結果を**表 3** に示す．

次に，千田方式においても復元の独立性を保つため，それぞれの秘密情報に独立にシードを割り当てる場合を考える．この場合，システム全体で必要な記憶容量は以下のようになる．

$$|s|/k \times n \times r \times m + |s| \times (k-1) \times n \times r \times m \tag{15}$$

これよりこの場合に提案方式に対する各秘密情報に独立にシードを割り当てた場合の千田方式の記憶容量の比較を行った場合，以下のようになる．

$$\frac{\frac{l}{t} + r \times m \times (n-k+1)}{n \times r \times m \times (\frac{1}{k} + k - 1) + \frac{n}{t}} \tag{16}$$

ここで，この場合の千田方式に対しても前述と同じ条件でそれぞれの n および k に対する具体的な容量の比較を行った結果を**表 4** に示す．

これより，提案方式は千田方式に対しても前述の Kr93 の方式に対してと同様，復元の独立性を保つ場合には大きな容量削減効果を持つ．

これらの結果より，提案方式は給与情報や，医療データなどそれぞれのデータサイズが比較的小さいデータの分散について，これらのデータを第三者に公開する場合を想定し，復元の独立性を保つとき，従来の方式に対して大きな容量削減効果を持つことが分かる．

5.2 安全性

本章では，提案方式の安全性について評価を行う．提案方式では鍵サーバとデータサーバという 2 通りのサーバが存在する．ここで，鍵サーバは自身の分散情報を擬似乱数を用いて生成しており，データサーバは前述の鍵サーバの擬似乱数を用いて定められた分散式から生成された分散情報を保持している．ここでは提案方式における安全性は鍵サーバが擬似乱数を生成する際に利用する暗号装置の安全性に依存するということを示す．ここで，前提として提案方式においてユーザおよびサーバ間の通信路は暗号化されているものとする．

提案方式において，攻撃者 A はある秘密情報 s_{ij} を復元するための情報を持つものとする．つまり攻撃者 A は秘密情報 s_{ij} に関する合計 k 個の擬似乱数 q_{ij} および分散情報 W_{ij} を持つ．ここでは特に，攻撃者 A は $x_1, \cdots, x_l$ というサーバ識別子を持つ鍵サーバから分散された擬似乱数 $q_{i1}, \cdots, q_{il}$ および $x_{l+1}, \cdots, x_k$ というサーバ識別子を持つデータサーバから送信された分散情報 $W_{il+1}, \cdots, W_{ik}$ という k 個の分散情報から秘密情報 s_{ij} の復元を行ったとする．また A は s_{i+1j} に関する分散情報を $k-1$ 個持っているとする．A はこれらの情報を用いて提案方式によって分散された秘密情報 s_{i+1j} を復元するため，攻撃を行う．このとき，A の攻撃が成功するとは，閾値である k 個よりも少ない分散情報から秘密情報 s_{i+1j} を復元することを意味する．ここで，以下の定理が成立する．

定理 1 提案方式を解くことができる可能性と提案方式において利用される暗号装置が解くことができる可能性は等価である．

証明：A を提案方式において閾値よりも少ない分散情報から秘密情報を復元しようとする攻撃者とする．B を提案方式において利用する暗号装置において暗号鍵を持つことなく秘密情報 s_{i+1j} に関する鍵サーバの生成する擬似乱数を生成しようとする攻撃者とするとき，A の攻撃が成功するならば，B の攻撃が成功することを示す．

A は仮定より $k-1$ 個の分散情報から秘密情報 $s_{i+1,j}$ を復元することが可能である．これより A は式 (17) における $k-1$ 個の係数を決定することができるため，式を一意に定めることができる．ここで，A は定められた式 (17) を B に対して送信する．

$$f(x) = s_{i+1j} + a_{i+11}x + \cdots + a_{i+1k-1}x^{k-1} \tag{17}$$

分散式を受け取った B は式中の x に対して任意の鍵サーバの識別子 x_{j_l} を入力することで擬似乱数 $q_{ij_l} = f(x_{j_l})$ を求めることができる．これにより，B は各鍵サーバの持つ鍵情報をいっさい知ることなく，公開情報であるデータ識別子 $dID[s_{i+1j}]$ を鍵サーバ x_{j_l} の鍵情報を用いて暗号化することで得られる擬似乱数 q_{i+1j_l} を求めることが可能となる．つまり，本提案方式において閾値未満の分散情報より分散式を決定することができる場合，利用する暗号装置において鍵情報を知ることなく，平文と暗号文の組合せを得ることができるため B の攻撃が成功する．

（QED）

以上により提案方式において利用する暗号装置から生成される擬似乱数の予想が困難であれば，提案方式における分散式の係数を予測することは困難である．つまり，提案方式において利用する暗号装置が計算量的安全性を持つ場合には，提案方式も同様に計算量的安全性を持つ．

5.3 提案方式による構成システムの特徴

本論文ではユーザの秘密情報 s_i に関する分散情報の分散先をすべて同等の記憶容量を持つサーバであるとしたが，提案方式には分散情報を削減し，鍵情報 key_i のみを持つ鍵サーバとすべての秘密情報 m 個に関する分散情報を持つデータサーバが存在している．

ここで，鍵サーバについては単なる暗号装置で構成することができる．具体的には，これらの鍵サーバはユーザから送信されたユーザ識別子 $ID[y]$ を自身の持つ鍵 key_j を用いて暗号化し，$Eid(y,j) = Enc(ID[y], key_j)$ を生成し，ユーザに送り返す．これを受け取ったユーザが自身の持つ秘密情報に割り振られた $dID[s(i)]$ を用いて擬似乱数 $q_{ij} = Enc(dID[s(i)], Eid(y,j))$ の生成を暗号装置を用いて行うため，鍵サーバが持つ情報は key_j のみであり，何人のユーザがいくつの秘密情報を分散させてもこのデータ量が増加することがない．これより，これらのサーバは実質的には送信された $ID[y]$ を暗号化するのみでよく，このような装置はデータの格納容量を持つサーバに比べ，はるかに安価かつ簡単に構成可能である．

これらすべてをこのような装置で構成することで，すべてデータサーバで構成する従来方式を用いて構成するクラウドシステムよりもより安価に大規模なクラウドシステムを構成することができる．

また，提案方式ではユーザが鍵サーバから送られてきた $Eid(y,j)$ を用いて擬似乱数 $q_{ij} = Enc(dID[s(i)], Eid(y,j))$ を自身の持つ暗号装置を用いて生成した．このように行うことでユーザは鍵サーバに自身のユーザ識別子 $ID[y]$ を送信するのみでよいため通信量の削減を行うことができる．しかし，このような処理ではユーザは暗号装置を持つ必要があり，暗号化処理を行うなど負荷が大きい．これに対して，ユーザが鍵サーバに直接データ識別子 $dID[s(i)]$ を送信し，鍵サーバは受け取った $dID[s(i)]$ と自身の持つ鍵 key_j を用いて暗号化して，擬似乱数 $q_{ij} = Enc(dID[s(i)], key_j)$ を生成することでも提案方式を構成することができる．この場合，鍵サーバは生成した m 個の擬似乱数 $q_{1j}, \cdots, q_{mj}$ をユーザに送信する必要があるため，通信量が増加するという欠点があるが，ユーザは暗号化装置を持つ必要がなく，負荷が軽減され，かつシステム構成を簡単化することができる．また，安全性についてもデータ ID$dID[s(i)]$ と秘密情報 s_i の間には式 (4) の関係が成立するため，この情報を鍵情報が受け取ったとしても提案方式と同等の安全性を保つといえる．

また，提案方式では計算量的に安全な擬似乱数を用いているため，ある分散情報を知りえても，他の秘密情報に関する分散情報については多項式時間では計算することができない．このため，復元の独立性を実現することができる．提案方式の特徴は，記憶容量削減と計算量的安全性を実現する他の方式，すなわち Kr93 や千田方式と異なり，記憶容量の増加なしに復元の独立性を実現する点といえる．

6. まとめ

本論文では以下の特徴を満たすクラウドコンピューティングに適した秘密分散法の提案を行った．

- 分散情報を持つサーバの記憶容量削減効果を持つ．
- 少なくとも計算量的安全性を持つ．
- 「復元の独立性」を持つ．

本提案方式を利用することで，最大 $k-1$ 台のサーバの持つべき情報を鍵情報のみにすることができる．これにより，従来方式に比べシステム全体で分散データ保管に関するコストを削減することができ，また，計算量的安全性を持つ暗号化装置を利用することでそれぞれの秘密情報について計算量的安全性を保つことができる．

謝辞 本論文作成にあたり，ご協力いただいた植松祐基さん，馬場雪乃さん，高荒亮さん，稲村勝樹さん，須賀祐治さんに心から感謝いたします．また，有益な助言や，アイディアをくださった東京理科大学岩村研究室の皆様に心から感謝いたします．

参考文献

[1] Mell, P. and Grance, T.: NIST によるクラウドコンピューティングの定義，NIST (2011).
[2] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. and Zaharia, M.: A view of cloud computing, *Comm. ACM* (2010).
[3] Shamir, A.: How to share a secret, *Comm. ACM*, Vol.22, No.11, pp.612–613 (1979).
[4] 山本博資：(k, L, n) しきい値秘密分散システム，電子通信学会論文誌，Vol.J68-A, No.9, pp.945–952 (1985).
[5] Blakley, G.R.: Security of ramp schemes, *Crypto'84*, pp.242–268 (1984).
[6] Krawczyk, H.: Secret Sharing Made Short, *Crypto'93*,

pp.136–146 (1994).
[7] 千田浩司，五十嵐大，濱田浩気，菊池 亮，冨士 仁，高橋克巳：マルチパーティ計算に適用可能な計算量的ショート秘密分散，*SCIS2012* (2012).
[8] 千田浩司，五十嵐大，菊池 亮，濱田浩気：計算量的秘密分散およびランプ型秘密分散のマルチパーティ計算拡張，情報処理学会研究報告書 (2012).
[9] Rabin, M.O.: Efficient dispersal of information for security, load balancing, and fault tolerance, *Journal of the Association for Computing Machinery*, pp.335–348 (1989).
[10] 小林士郎，岩村恵市：マルチパーティ計算に適用可能な計算量的秘密分散に関する評価，*CSS2012*, pp.60–67 (2012).
[11] 高橋 慧，岩村恵市：クラウドコンピューティングに適した計算量的安全性を持つ秘密分散法，*CSS2012*, pp.52–59 (2012).
[12] 山本博資：秘密分散法とそのバリエーション，数理解析研究所講究録，Vol.1361, pp.19–31 (2004).
[13] Jackson, W. and Martin, K.: A Combinatorial Interpretation of Ramp Schemes, *Australasian Journal of Combinatorics*, Vol.14, pp.51–60 (1996).
[14] 尾形わかは：多数の秘密の分散に適する計算理論的検証可能秘密分散共有法，*SCIS2012* (2012).
[15] Kurosawa, K., Okada, K., Sakano, K., Ogata, W. and Tsujii, S.: *Nonperfect Secret Sharing Schemes and Matroids*, pp.126–141, Springer-Verlag (1998).
[16] 蓑輪 正：大容量データに向けたセキュア分散ストレージシステム，*SCIS2012* (2012).
[17] 山本 剛，小林鉄太郎，山本具英，冨士 仁，高橋克巳：暗号技術はクラウドで情報を保護できるか，*SCIS2012* (2012).
[18] 堀内公平：MyCloud：複数ベンダのクラウドを用いて構成する秘密分散ストレージ，プログラミングシンポジウム，pp.161–164 (2010).
[19] Bai, L.: A Strong Ramp Secret Sharing Scheme Using Matrix Projection, *WoWMoM'06* (2006).
[20] Ben-Or, M., Goldwasser, S. and Wigderson, A.: *Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation*, ACM (1988).
[21] Harn, L.: Efficient sharing (brodcasting) of multiple secrets, *IEE TECHNICAL NOTE*, pp.237–240 (1995).
[22] Halpern, J. and Teague, V.: Rational Secret Sharing and Multiparty Computation: Extended Abstract, *STOC'04* (2004).
[23] Blundo, C., Cresti, A., De Santis, A. and Vaccaro, U.: Fully Dynamic Secret Sharing Scheme, *Crypto'93*, pp.110–125 (1994).

高橋 慧

平成 24 年東京理科大学電気工学科卒業．平成 24 年同大学大学院修士課程進学．

小林 士郎

平成 24 年東京理科大学工学部電気工学科卒業．平成 24 年同大学大学院修士課程進学．

岩村 恵市 （フェロー）

昭和 55 年九州大学工学部情報工学科卒業．昭和 57 年同大学大学院修士課程修了．同年キヤノン株式会社入社．平成 6 年東京大学工学博士．現在，東京理科大学工学部電気工学科教授．主に符号理論，並列処理，情報セキュリティ，電子透かしの研究に従事．IEEE，電子情報通信学会，情報ハイディングおよびその評価基準（IHC）研究会委員長．