マルチパーティ計算に適用可能な計算量的秘密分散法に関する評価

小林　士郎†　　岩村 惠市†

†東京理科大学
102-0073　東京都千代田区九段北 1-14-6
kobayashi @sec.ee.kagu.tus.ac.jp, iwamura@ee.kagu.tus.ac.jp

あらまし Shamirの秘密分散法は分散情報を復元せずにデータ処理が行えるマルチパーティ計算に拡張可能だが記憶容量が元情報のn倍となるため効率が悪い．それに対し千田らによって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が提案されている．この手法は分散情報の記憶容量を削減でき計算量的安全性を実現する．一方、SCIS2010では植松・岩村によって同様の秘密分散法が提案されている．さらにCSS2012にて高橋・岩村によって植松・岩村による方式を改良した手法が提案されている．本論文は3つの手法に加えH.Krawczykの手法を加えた4方式を比較し記憶容量、計算量、通信量などを評価する．

Evaluation of computational secret sharing schemes that can be applied to multi-party computation

Shiro Kobayashi†　Keiichi Iwamura†

†Tokyo University of Science
1-14-6 Kudankita, Chiyoda, Tokyo 102-0073, JAPAN
{kobayashi, iwamura}@sec.ee.kagu.tus.ac.jp

Abstract Shamir’s secret sharing scheme can be extended to multi-party computation but inefficient because the capacity of information n times. In contrast, the short secret sharing computational applicable to multi-party computation has been proposed by Chida. This method can reduce the capacity to provide computational safety. Moreover, similar approach has been proposed by Iwamura, Uematsu in SCIS2010. And, the method by Takahashi, Iwamura at CSS2012, an improved method by Iwamura, Uematsu has been proposed. In addition to the method of Krawczyk, this paper evaluates the capacity, the amount of calculation, and the amount of communication compared to the four systems.

1　はじめに

近年、クラウドコンピューティングの普及によりクラウドに預ける情報の秘匿性、耐消失性を高める秘密分散が注目されている．その中で代表的な Shamir 秘密分散[1]は n 個に分散された分散情報のうち k-1 個以下の分散情報からは元の秘密情報を一切復元できないが、k 個集めれば復元できる手法である．しかし、Shamir 秘密分散の問題点として分散情報を保管するサーバへの容量効率の悪さが挙げられる．

そこでサーバの容量効率を改善でき、かつマルチパーティ計算を実現する秘密分散法を考える．その１つとして近年、千田らによって提案されている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(以下千田方式とする)[2]がある. この手法は分散情報をIDA により記憶容量を削減させる．さらに分散情報を Shamir 秘密分散の分散情報に変換することによりマルチパーティ計算に適用可能にする方式である.

一方 SCIS2010 において植松・岩村によるマルチパーティ計算に適用可能であり計算量的安全性を持つ秘密分散法（以下植松方式とする）[4]が提案されている．この方式は秘密情報が多数あることを前提としている方式であり分散情報と擬似乱数が同じになるように分散式内の係数を算出し擬似乱数を共通化させることによって記憶容量の削減を図っている. さらに CSS2012 において高橋・岩村によって植松・岩村による方式を改良した手法（以下高橋方式とする）[5]が提案されている.

そこで本論文ではこの３つの手法[2][4][5]に加えて Krawczyk らによる秘密分散法（以下 Kr93 方式とする）[6]を加えた４つの方式を比較し、その記憶容量、計算量、通信量を評価する．また、比較する際に様々なパラメータを考慮しなければならない．よってマルチパーティ計算を行う上でのクラウドコンピューティングにおける実用性の観点から分散数、閾値、秘密情報数、扱うデータサイズなどを考察していく.

2　準備

４方式の比較を行う前に前提条件を以下のように決める.

①クラウドを想定するため q 人のユーザが、各々m 個の秘密情報を分散させ、そして同じく m 個を独立に復元させる場合を考える.

②複数のユーザ(q 人)と複数のデータ(m 個)を用いるので、サーバがわかるようにユーザの識別子としての ID とデータ識別子の dID を持つ(簡単のため|ID|=|dID|として扱う).

③分散は m 個のデータ一括で行うが復元においては m 個のデータを一括して復元するのではなく各秘密情報を独立に復元(m 回)する.

④秘密情報はユーザが持ちサーバに秘密情報を知らせずに秘匿計算するものとする.

⑤記憶容量に関しては n 個の全サーバにある分散情報の量で評価する．また、ユーザやサーバが持つ ID は記憶容量に含めない.

上記①～⑤の前提条件を踏まえうえで各項目を評価する.

次に各４方式の比較に用いる数式、記号を以下のように定める.

・C　真性乱数(鍵)を生成する際の計算量
・G　擬似乱数を生成する際の計算量
・S(X)　秘密情報 X を分散する際の計算量
・L(X)　X を復元する際に用いる Lagrange 多項式計算の計算量
・H(X)　X を暗号化（復号）する際の計算量
・T(X)　X を送信する際の通信量

3　４方式について

ここでは比較する４方式の分散処理、復元処理を記し、その際計算量、通信量、記憶容量を示す.

3.1　千田方式[2]

千田方式の分散処理、復元処理を以下に記す.

3.1.1　分散処理

① ユーザは自分の ID と m 個のデータ ID を持つ.
② ユーザは真性乱数 R を k-1 個用意する.
③ ユーザは k-1 個の R を n 個のサーバに分散させる。同時にサーバに自分の ID も送る.
④ ユーザは k-1 個の真性乱数を初期値として m 個のデータに対して擬似乱数 V(R) を生成し、サーバへ送る.
⑤ ユーザは秘密情報と k-1 個の V(R)を多項式の係数として秘密情報毎に分散値 h(k)

を生成する．
⑥ ユーザは m 個の秘密情報に対する h(k)を IDA を用いて n 個のサーバに分散記憶させる。ただしデータ ID は m 個のデータの送り順とするため送らない．データ ID は復元時に用いられる．
⑦ q 人のユーザが独立に行う．

3.1.2 復元処理

① ユーザは自分の ID をサーバに送信する．サーバはユーザに乱数の分散情報を送る．
② k-1 個の乱数 R を復元する．
③ ユーザは送られてきた分散情報から復元した R を用いて擬似乱数 V(R)を生成する．
④ ユーザは復元したいデータ ID を指定して、k 個のサーバから h(k)の分散情報を集める．
⑤ ユーザは IDA の復元処理より h(k)を復元する．
⑥ ユーザは復元された h(k)と擬似乱数を用いて秘密情報 S を復元する．
⑦ q 人のユーザが独立に m 個のデータについて行う．

3.1.3 計算量

・分散処理
② C×(k-1)×q
③ S(R)×(k-1) ×n×q
④ G×(k-1)×m×q
⑤ S(S)×m×q
⑥ S(h(k))×m×n×q
・復元処理
② L(R)×(k-1) ×m×q
③ G×(k-1) ×m×q
⑤ L(h(k))×m×q
⑥ L(S)×m×q

3.1.4 通信量

・分散処理
③ T(r)×n×(k-1)×q+ T(ID) ×n×q
④ T(V(R))×(k-1)×m×q
⑥ T(h(k))×n×m/k×q
・復元処理
① T(ID)×m×k×q + T(r)×k×(k-1)×m×q
④ T(h(k))/k×k×m×q+T(ID)×k×m×q

3.1.5 記憶容量

サーバに記憶されるのは q 人分の乱数の分散情報と h(k)に関する分散情報になる．h(k)は IDA で分散されるので 1/k 倍される．
記憶容量=|S|/k×n×m×q＋|r|×(k-1)×n×q

3.2 植松方式[4]

植松方式の分散処理、復元処理を以下に記す．

3.2.1 分散処理

① k-1 個の各サーバは自分の鍵を持ち、ユーザは自分の ID と m 個のデータ ID を持つ．
② ユーザは自分の ID をサーバに送信し、1 つ目の秘密情報を分散させる．
③ k-1 個のサーバは分散情報を初期値として鍵 Key を用いて擬似乱数 V(R)を m-1 個生成し、ユーザへ送る．ここで、データ ID はデータの送り順とするため、dID は送らない．
④ ユーザは分散情報＝擬似乱数となるように分散式内の係数を算出する．
⑤ n-(k-1)個のサーバが持てばよい⑤の係数を用いた分散情報を生成する．
⑥ q 人のユーザが独立に m 個のデータについて行う．

3.2.2 復元処理

① ユーザは自分の ID と復元するデータ ID をサーバに送信する．
② k-1 個のサーバは Key と一つ目の分散情報を用いて擬似乱数を生成する．
③ k 個のサーバは分散情報をユーザに送る．
④ ユーザは秘密情報を m 個復元する．
⑤ q 人のユーザが独立に m 個のデータについて行う．

3.2.3 計算量

・分散処理
① C×(k-1)×q
② S(S)×n×q

③ G×(k-1)×(m-1)×q
④ L(V(R))×(m-1) ×q
⑤ S(S)×(n-k+1)×(m-1)×q
・復元処理
② G×(k-1)×m×q
③ L(S)×m×q

3.2.4　通信量
・分散処理
② T(ID) ×n×q+T(S)×n×q
③ T(V(R))×(k-1)×(m-1)×q
⑤ T(S) ×(n-k+1)×(m-1)×q
・復元処理
① 2×T(ID) ×k×m×q
③ T(S)×k×m×q

3.2.5　記憶容量
k-1 個のサーバに保管されるのは 1 つの分散情報と鍵 Key であり、残りのサーバには m 個の秘密情報分の分散情報である.
記憶容量=(|S|+|Key|)×(k-1) ×q＋
|S|×(n-(k-1))×m×q

3.3　高橋方式[5]
3.3.1　分散処理
① k-1 個のサーバは自分の鍵を持つ.
② ユーザは自分の ID と m 個の秘密情報に対する dID を持つ.
③ ユーザは自分の ID を k-1 個のサーバに送る.
④ k-1 個のサーバは送られてきた ID を鍵 Key を用いて暗号化する.
⑤ k-1 個のサーバはその暗号結果をユーザに返す.
⑥ ユーザは送られてきた暗号結果を鍵として m 個のデータ ID を暗号化する.
⑦ ユーザはそのデータ暗号結果を用いて分散情報＝データ暗号結果となるように分散式内の係数を算出する.
⑧ n-(k-1)個のサーバに⑤の係数を用いて分散情報を生成し n-(k-1)個のサーバに送る.
⑨ q 人のユーザが独立に m 個のデータについて行う.

3.3.2　復元処理
① ユーザは自分の ID と m 個のデータ ID を n 個のサーバに送る.
② k-1 個のサーバはユーザ ID と m 個のデータ ID を鍵 Key を用いて暗号化しデータ暗号結果を生成する.
③ n 個のサーバはそれぞれ持つ分散情報と疑似乱数をユーザに送る.
④ ユーザは送られてきた分散情報と疑似乱数を用いて Lagrange 多項式計算を行い秘密情報を復元する.
⑤ q 人のユーザが独立 m 個のデータについて行う.

3.3.3　計算量
・分散処理
① C×(k-1)×q
④ G×(k-1)×q
⑥ G×(k-1)×m×q
⑦ L(V(R))×m×q
⑧ S(S)×(n-k+1)×m×q
・復元処理
② C×(k-1)×m×q+G(k-1)×m×q
④ L(S)×m×q

3.3.4　通信量
・分散処理
② T(ID)×(k-1)×q
⑤ T(ID)×(k-1)×q
⑧ T(S)×(n-k+1)×m×q
・復元処理
① 2×T(ID)×k×m×q
③ T(S)×k×m×q

3.3.5　記憶容量
k-1 個のサーバに保管されるのは鍵 Key のみ、n-(k-1)個のサーバに保管されるのは Shamir 秘密分散で分散された分散情報のみとなる.

記憶容量=|Key|×(k-1) + |S|×(n-(k-1))×m×q

3.4　Kr93 方式[6]

Kr93 方式の分散処理、復元処理を以下に示す.

3.4.1　分散処理

①。ユーザは自分の ID と m 個の秘密情報に対する dID を持つ.
② ユーザは秘密情報毎に鍵 Key を用意する.
③ ユーザは鍵を用いて各データを暗号化し E(S)を生成する.
④ ユーザは E(S)をランプ型秘密分散を用いて分散記憶させ、自分の ID も送る.
⑤ ユーザは鍵を分散させる.
⑥ q 人のユーザが独立に m 個のデータについて行う.

3.4.2　復元処理

① ユーザは自分の ID と復元したいデータ ID を k 個のサーバに送る.
② k 個のサーバは指定されたデータの鍵と秘密情報の分散情報を送る.
③ ユーザは Key を復元する.
④ ユーザは暗号文 E(S)を復元する.
⑤ ユーザは復元した鍵を用いて秘密情報を復号する.
⑥ q 人のユーザが独立に m 個のデータについて行う.

3.4.3　計算量

・分散処理
② C×m×q
③ G×m×q
④ S(E(S))×n×m×q
⑤ S(Key)×n×m×q
・復元処理
③ L(Key)×m×q
④ L(E(S))×m×q
⑤ H(S)×m×q

3.4.4　通信量

・分散処理
① T(ID) ×n×q
③ T(E(S))/k×n× m×q + T(ID) ×n×m×q
④ T(S)/k×n×m×q+T(ID)×q
⑤ T(Key)×n×m×q
・復元処理
① 2T(ID) ×k×m×q
② T(Key)×k ×m×q+T(E(S))/k×k×m×q

3.4.5　記憶容量

サーバに保管されるのはランプ型秘密分散で分散された暗号化分散情報と鍵 Key の分散情報である.
記憶容量＝|S|/k×m×n×q＋|Key|×n× m×q

3.5　計算量・通信量・記憶容量まとめ

４方式の計算量・通信量・記憶容量を式で表し表にまとめたものを示す．また暗号化、復号計算は疑似乱数生成計算と同じ作業と見なすことができるので G=H(X)とする.
・分散処理

表１　分散処理における４方式の計算量

	擬似乱数生成	分散計算	Lagrange
千田	G(k-1)mq	S(R)n(k-1)q S(S)mq S(h(k))nmq	
植松	G(k-1)(m-1)q	S(S)nq S(S)(n-k+1)mq	L(V(R))(m-1)q
高橋	G(k-1)mq G(k-1)q	S(S)(n-k+1)mq	L(V(R))(k-1)mq
Kr93	Gmq	S(S)nmq S(Key)nmq	

表２　復元処理における４方式の計算量

	擬似乱数生成	Lagrange
千田	G(k-1)mq	L(S)mq L(R)(k-1)mq L(h(k))mq
植松	G(k-1)(m+1)q	L(S)mq
高橋	G(k-1)mq	L(S)mq
Kr93	Gmq	L(Key)mq L(S)mq

表３　４方式の通信量

	通信量（分散）	通信量（復元）
千田	T(r)n(k-1)q T(h(k))nmq/k T(ID) nq T(V(R))(k-1)mq	T(r)k(k-1)mq T(h(k))mq 2T(ID)kmq

植松	T(S)nq+ T(ID) nq T(S) (n-k+1)(m-1)q T(V(R))(k-1)(m-1)q	T(S)kmq 2T(ID) kmq
高橋	2T(ID)(k-1)q T(S)(n-k+1)mq	2T(ID)kmq T(S)kmq
Kr93	T(S)nmq/k T(Key)nmq T(ID) q(1+n) T(ID) nmq	T(Key)kmq T(S)mq 2T(ID) kmq

表４　４方式の記憶容量

	記憶容量
千田	|S|/k*nmq＋|r|(k-1)nq
植松	(|S|+|Key|)(k-1)q+|S|(n-k+1)mq
高橋	|Key|(k-1) + |S|(n-k+1)mq
Kr93	|S|/k*nmq+|Key|nmq

各項目多くのパラメータが存在し、比較することは困難であるので、4 章でパラメータに関する考察を行い比較する.

4　パラメータの考察

3.5 節において式で示したが、比較が容易ではないのでクラウドコンピューティングにおけるマルチパーティ計算を実用性の観点から考慮したパラメータを考察し、設定する.

・分散数 n

分散させる意味を考えると n>2 が必要になる．さらに n を大きくすれば攻撃耐性やサーバ欠損耐性は強くなるがサーバ設置場所の確保、設備設置費用、メンテナンス費も同時に膨大になるので n≧5 は考えにくい. よって今回は n が 3,4 程度になることを想定する.

・閾値 k

n-k が大きいとサーバ欠損耐性は強くなるが適切なメンテナンスが行われる場合 2 台以上同時に使用不能になる可能性は考えにくい. k が小さいと攻撃耐性が低いので大きいほうが良い．よって今回は n=3,4 なので (n,k)=(3,2),(4,3),(4,2)を考える.

・秘密情報データサイズ|S|

データが画像や文書の場合|S|は鍵サイズ|Key|に比べて十分に大きいと考えられる．しかしマルチパーティ計算を考えた場合|S|は個人に関連する数字となるので大きいサイズになることは考えがたい．仮に 128 ビット共通鍵暗号を今回の 4 方式で用いるとすれば |Key|=128 とできる．よって|S|=t|Key|とすれば t<1 となる。今回は t=0.5,1 の場合を考える.

・秘密情報数 m

クラウドコンピューティングにおけるマルチパーティ計算を考えた場合、m は比較的大きな数と考えられる．ここでは一応、m=1000000 とする.

・乱数または鍵サイズとユーザ数 q

初期値として真性乱数を用いるので乱数は小さいサイズであり 80 ビット以上と考えられる．そこで簡単のため Key と同サイズの |R|=128 とする．よって|Key|=|R|となる.

また、1 つの乱数または暗号鍵を複数の秘密情報に対して使いまわすと、秘密情報が十分大きい場合、乱数または鍵を無視できる．しかし、ユーザが異なる場合、同じ乱数または鍵を使いまわすことはできない．高橋方式は異なるユーザに対してもサーバは 1 つの鍵を管理するだけでよいが、他の方式はそうではない．よって q=100,10000 と評価する.

・ID について

ID の大きさを考えると非常に小さいものと考えられるが ID と秘密情報は対応するものであり、小さすぎると安全性面で問題がある．そこで今回は鍵サイズ|Key|と同程度のサイズとし|Key|=|ID|(=|dID|)とする.

以上決定した事項を用いて計算量、通信量、記憶容量を比較する.

5　各項目比較

3.5 節で示した表と 4 章で述べたパラメータを踏まえて各項目比較検討する.

5.1　計算量

計算量は表 1,2 の合計値を考える．擬似乱数生成、分散計算、Lagrange 多項式計算と分けて比較する.

5.1.1　擬似乱数生成計算量

・千田方式

2G(k-1)mq

・植松方式
2G(k-1)mq
・高橋方式
2G(k-1)(2m+1)
・Kr93 方式
Gmq+Gmq=G2qm
少ない方から並べると Kr93 方式＜千田方式＝植松方式≒高橋方式となる.

5.1.2 Lagrange 多項式計算量

擬似乱数計算量と同じく表 1,2 の合計値を考えるが Lagrange 多項式計算を行う回数を考えれば比較が容易になる.
・千田方式
mq+(k-1)mq+mq=mq(k+1)
・植松方式
(k-1)(m-1)q+mq≒mqk
・高橋方式
(k-1)mq+mq=mqk
・Kr93 方式
mq+mq=2mq
少ない方から並べると Kr93 方式≒高橋方式=植松方式＜千田方式となる. 千田方式は復元する時に乱数、分散情報、秘密情報と他方式よりも Lagrange 多項式計算を行う回数が多いからと考えられる.

5.1.3 分散計算量

・千田方式
表より式を評価し、計算量を W とすると、
W=S(Key)q{n(k-1)+tm+tmn} (1)
・植松方式
表より式を評価し、計算量を X とすると、
X=S(Key)qt(n+(n-k+1)m) (2)
・高橋方式
表より式を評価すし、計算量を Y とすると、
Y=S(Key)(n-k+1)tmq (3)
・Kr93 方式
表より式を評価すし、計算量を Z とすると
Z＝S(Key)mnq(t+1) (4)
以上(1)~(4)の式に決定したパラメータを代入し計算した結果、少ないほうから並べると高橋方式＜植松方式＜千田方式＜Kr93 方式となった. 植松方式や高橋方式は秘密情報をただ分散させるのではなく擬似乱数や暗号化した ID が分散情報と同じになるように計算するので少なくなると考えられる.

以上より計算量において Lagrange 多項式計算が支配的であり、ついで擬似乱数計算、分散計算とすると全体的に高橋方式、Kr93 方式、植松方式、千田方式の順で計算量が小さくなると考えられる.

5.2 通信量

こちらも同様に考察したパラメータを考慮して 4 方式の式を変形し比較しやすくする.
・千田方式
表より式を評価し、通信量を W とすると、
W=T(Key)q{m(tn/k+2k-1+k^2+t)+nk} (5)
・植松方式
表より式を評価し、通信量を X とすると、
X＝T(Key)q{n(t+1)+(m-1)n+km(t+2)} (6)
・高橋方式
表より式を評価し、通信量を Y とすると、
Y=T(Key)q{m(2k+t(n-k+1)+tk)+2(k-1)} (7)
・Kr93 方式
表より式を評価し、通信量を Z とすると、
Z=T(Key)q{m(nt/k+3k+2n+t)+1+n} (8)
以上(5)~(8)の式に決定したパラメータを代入し計算した結果、少ないほうから並べると高橋方式＜植松方式≒千田方式＜Kr93 方式となった. 通信量はユーザとサーバ間でデータを交換する量であるが、扱うデータの大きさだけではなく通信する回数にも依存する. データ ID やユーザ ID を通信する回数を省略できる高橋方式が優位になると考えられる.

5.3 記憶容量

こちらも同様に考察したパラメータを考慮して 4 方式の式を変形し比較しやすくする.
・千田方式
表より、|S|/k*nmq＋|r|(k-1)nq なので記憶容量を W とすると、
W=|Key|nq{tm/k+(k-1)} (9)
・植松方式
表より、(|S|+|Key|)(k-1)q+|S|(n-(k-1))mq なので

記憶容量を X とすると、
X=|Key|q{(t+1)(k-1)+tm(n-k+1)}　　(10)

・高橋方式
表より、|Key|(k-1) + |S|(n-(k-1))mq なので記憶容量を Y とすると、
Y=|Key|{(k-1)+tmq(n-k+1)}　　(11)
・Kr93 方式
表より、|S|/k*nmq+|Key|nmq なので記憶容量を Z とすると、
Z =|Key|nmq(t/k+1)　　(12)
以上(9)~(12)の式にの式に決定したパラメータを代入し計算した結果、少ないほうから並べると千田方式＜高橋方式＜植松方式＜Kr93 方式となる．ここで、千田方式と高橋方式の式を細かく見ると、
千田方式：|S|/k×n×m×q＋|r|×(k-1)×n×q
高橋方式：|Key|×(k-1) + |S|×(n-(k-1))×m×q
(n,k)=(4,3)のとき
千田方式：|S|/3×4×m×q＋|r|×8×q
=|Key|(mqt3/4+8q)
高橋方式：|Key|×2+ |S|×2×m×q
=|Key|(2+2mqt)
mqt3/4+8q>2+mq となる m,q を考えると、2<mq/4+8q=q(8-m/4)より、m<32 であれば任意の q について高橋方式が優位になる．今回は m=1000000 としているが、例えば 1 人患者に対して 32 項目以下の情報による統計値を計算する場合等を考えると高橋方式が良いことになる．さらに今回考えていないパラメータとして n を比較的大きくとれて n≒k となる場合をシミュレーションした結果、全パターンにおいて高橋方式が優位となった．このようにサーバ数 n や閾値 k、情報数 m によって優位な方式が変化することが考えられる．

5.4　千田方式の IDA 利用について
本論文では千田方式は分散情報 h(k)を分散する際 IDA を用いている．ただ[3]では、IDA を用いずに Kr93 方式を用いて分散させている．Kr93 方式を用いた場合を想定してシミュレーションした結果、記憶容量において鍵の分散情報が m×q 個分増えるので必然的に記憶容量も増え、高橋方式＜植松方式＜千田方式となった．

6　まとめ

本論文では、マルチパーティ計算を実用性の観点から 4 方式の計算量、通信量、記憶容量を比較・評価した．高橋方式が通信量、計算量において優位であったが、記憶容量は千田方式が優位であった．しかし 5.3 節のようにパラメータが設定されたり、n と k が近い値であれば、高橋方式が優位になる場合もある．

7　参考文献

[1]Shamir,A :How to share a secret. ACM22(11),pp612-613(1979)
[2]千田浩司,五十嵐大,濱田浩気,菊池亮,冨士仁,高橋克己 :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
[3]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
[4]植松祐基,岩村恵市 :複数の情報を有するメモリやデータベースに適した秘密分散法 SCIS2011.
[5]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
[6]Krawczyk,H :Secret shareing made short. CRYPTO1993,pp136-146