SCIS 2014 The 31st Symposium on
Cryptography and Information Security
Kagoshima, Japan, Jan. 21 - 24, 2014
The Institute of Electronics,
Information and Communication Engineers

暗号処理と可換な (2,n) 閾値秘密分散法とその秘密計算法への適用
A (2,n)-Threshold Secret Sharing Scheme with the Commutative Property and Its Application to Secure Computations

倉上 高史 * 藤吉 正明 * 貴家 仁志 *
Takashi Kurakami Masaaki Fujiyoshi Hitoshi Kiya

あらまし 本稿では，算術加算演算に基づく，暗号処理との可換性を有する (2,n) 閾値秘密分散法を提案する．さらに，提案手法が秘密計算に適用可能であることを示す．可換性および秘密計算可能という特徴を有する提案法は，クラウドサービスへの適用が期待される．排他的論理和演算による秘密分散法は，可換性を持つが秘密計算が実現できなかった．一方，従来の秘密計算可能な秘密分散は，一般に可換性を有しない．

キーワード プライバシー保護，クラウドコンピューティング，算術加算，乱数加算，秘匿関数計算

1 はじめに

近年，秘密分散技術をクラウドサービスに用いることで，安全なクラウド環境を実現する動きがある．そこで，クラウド事業者が不正した場合などを想定し，秘密分散の上にさらに暗号化や階層的な秘密分散を行うモデルを考える必要がある．文献 [1] では，暗号化および秘密分散を施した場合，秘密情報を復元するデータフローを考察している．これによると，排他的倫理和の可換性により，排他的論理和演算のみで構成された秘密分散法 [2]–[7] は，排他的論理和演算で実現される暗号処理に対して可換である．この可換性により，複数のクラウド事業者を介したときに，秘密分散 (暗号化) の順序は復元 (復号化) の順序を限定しない，非対称なクラウドサービスを実現可能であるとしている．また，排他的論理和演算は高速で実現できるため，実システムに適しているといえる．

演算速度および暗号・秘密分散の間の可換性という観点から，クラウドサービスに秘密分散法を活用する場合，排他的論理和演算で構成される秘密分散法が有効であると考えられる．しかし，これらの秘密分散技術ではクラウドコンピューティングで需要がある秘密計算を実現することができない．

秘密計算とは，それぞれ分散情報を持つ複数の主体による協調計算で秘密情報を秘匿しつつ計算結果を求める手法である．従来の秘密計算可能な秘密分散法では処理効率に問題があったが，千田らは，秘密計算を算術演算と論理回路演算に分けることで，処理効率を向上させた方式を提案している [8]．$(2,3)$ 閾値法に限定されていた千田らの方式を，さらに布川らは，(k,n) 閾値法に拡張している [9]．しかしこれらの方式は，排他的論理和で構成された秘密分散法が持つ可換性を有しない．

本稿では，可換な演算として算術加算演算を用いることで，暗号処理との間に可換性を有し，さらに秘密計算が実現可能な新しい $(2,n)$ 閾値秘密分散法を提案する．また，暗号化した分散情報による秘密計算が実行可能であることをを示す．さらに，提案法は，Shamir の (k,n) 閾値法 [10] のような多項式による構成でないため，マルチパーティ計算で積を求める際，次元削減およびランダム化処理を必要としないという特徴を有し，相互通信を必要としないことが期待される．

2 準備

本節では，排他的論理和演算で構成される秘密分散法を紹介し，その暗号処理との間の可換性を確認する．また，秘密計算の概要を示し，その要件を確認する．

2.1 排他的論理和演算を用いた秘密分散法

藤井らの方式 [2] を例にして，排他的論理和演算のみで実現される秘密分散法を説明する．ただし以下では簡単のために $(2,3)$ 秘密分散とする．

(a) 分散処理

まず，秘密情報 S を $S = s_1 \| s_2$ $(s_i = \{0,1\}^d)$ として部

* 首都大学東京システムデザイン研究科, 〒 191–0065 東京都日野市旭が丘 6–6, Dept. of Information and Communication Systems Engineering, Tokyo Metropolitan University, 6–6 Asahigaoka, Hino–shi, Tokyo, 191–0065 Japan

表 1: 排他的論理和による (2, 3) 閾値法.

W_i	$w_{(i,0)} \parallel w_{(i,1)}$	
W_0	$s_0 \oplus R_0$	$s_2 \oplus R_1$
W_1	$s_1 \oplus R_0$	$s_0 \oplus R_1$
W_2	$s_2 \oplus R_0$	$s_1 \oplus R_1$

分秘密情報 s_1, s_2 に分割し，データ長 d ビットの独立な乱数 R_1, R_2 を準備する．ここで $\parallel$ は連結を表す．次に，$s_0 \in \{0\}^d$ として，

$$w_{(i,m)} = s_{i-m \bmod 3} \oplus R_m \quad (0 \le i \le 2, 0 \le m \le 1) \tag{1}$$

のように部分秘密情報 s_i と乱数 R_m の排他的論理和を求めることで部分分散情報 $w_{(i,m)}$ を生成する．ただし $\oplus$ は排他的論理和を表す．さらにそれらを表 1 の組み合わせで連結して分散情報 W_i を生成する．

(b) 復元処理
まず，分散情報 W_i から任意の 2 個 $(W_i, W_j\ (i \neq j))$ を取り出す．$W_i = w_{(i,0)} \| w_{(i,1)}$ であるから，互いの部分分散情報の排他的論理和は，$w_{i1} \oplus w_{j1} \| w_{i2} \oplus w_{j1}$ と表すことができる．演算結果として s_1, s_2 がそれぞれ独立に出力されるので，それらを連結することで秘密情報 S が復元される．

2.2 暗号化・秘密分散の可換性

文献 [7] では，暗号・秘密分散処理の間の可換性は，暗号化および秘密分散の処理に共に排他的論理和演算のみを使うことで，排他的論理和演算の可換性を継承する形で実現されている．暗号化としてストリーム暗号やブロック暗号の CTR モードを用いることで，排他的論理和演算のみで暗号化および秘密分散が構成される．

ここで，秘密情報に対して暗号化と秘密分散を併用する場合，秘密分散を先に行うか，暗号化を先に行うかという問題が生じる．

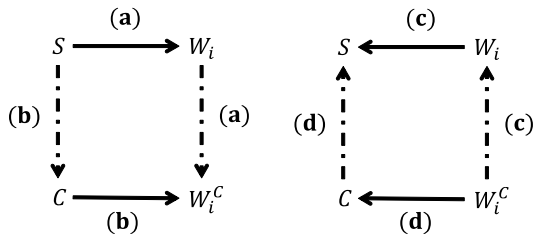
(a) (1) 秘密分散 →(2) 暗号化.
まず，式 (1) に従い秘密情報 S を秘密分散し，さらに分散情報 W_i を暗号化する場合を考える．分散情報の暗号化は，秘密分散された表 1 の分散情報 W_i と秘密鍵 K_1, K_2 $(K_i = \{0,1\}^d)$ との間の排他的論理和演算によって，

$$w^C_{(i,m)} = w_{(i,m)} \oplus K_{i-m \bmod 3} \tag{2}$$

のように実行される．これにより部分秘密情報 s_i を含んだ部分分散情報 $w_{(i,m)}$ が暗号化され，表 2 のように暗号化された分散情報 W^C_i が得られる．

表 2: 排他的論理和による (2, 3) 閾値法 + 暗号化.

W^C_i	$w^c_{(i,0)} \parallel w^c_{(i,1)}$	
W^C_0	$s_0 \oplus R_0$	$s_2 \oplus R_1 \oplus K_2$
W^C_1	$s_1 \oplus R_0 \oplus K_1$	$s_0 \oplus R_1$
W^C_2	$s_2 \oplus R_0 \oplus K_2$	$s_1 \oplus R_1 \oplus K_1$



暗号化・秘密分散.　　復号化・復元.

図 1: 暗号・秘密分散の間の可換性 [7].

(b) (1) 暗号化 →(2) 秘密分散.
暗号文 C を $C = c_1 \| c_2$ とすると，c_1, c_2 は部分暗号文である．ただし，C は，部分秘密情報 s_1, s_2 に対して $c_1 = s_1 \oplus K_1, c_2 = s_2 \oplus K_2$ のようにそれぞれ暗号化し，それらを連結したものである．すなわち，

$$C = s_1 \oplus K_1 \| s_2 \oplus K_2 \tag{3}$$

となる．この C に対して秘密分散を実行すると，式 (1) と同様に，

$$w^c_{(i,m)} = c_{i-m \bmod 3} \oplus R_m \tag{4}$$

となり，表 2 となる．以上，(a) および (b) の場合を図 1(暗号化・秘密分散) に示す．秘密情報 S に対し暗号化および秘密分散を併用すると，暗号化された分散情報 W^C_i の生成は，排他的論理和の可換性から，暗号化および秘密分散の順序によらず一意に決まることが確認できる．

次に，暗号化された分散情報に対して復号化を先に行う場合と，復元を先に行う場合に分ける．

(c) (1) 復号化 →(2) 復元.
まず，暗号化された分散情報 W^C_i を復号化し，さらにその分散情報 W_i を復元する場合を考える．W^C_i の暗号化された部分分散情報 $w^c_{(i,m)}$ に対し，秘密鍵 K_1, K_2 を用いて復号化すると，

$$w_{(i,m)} = w^c_{(i,m)} \oplus K_{i-m \bmod 3} \tag{5}$$

となり，部分分散情報 $w_{(i,m)}$ が得られる．それらを連結すると W_i となり，復元処理を行えば S が出力される．

(c) (1) 復元 →(2) 復号化.
$w^c_{(i,m)}$ に対し復元処理を行うと，

$$C = s_1 \oplus K_1 \| s_2 \oplus K_2 \tag{6}$$

となり，暗号文 C を得る．部分暗号文 c_1, c_2 に対してそれぞれ秘密鍵 K_1, K_2 を用いて復号化すると，部分秘密情報 s_1, s_2 が得られ，それらを連結すると秘密情報 S となる．

以上，(c) および (d) の場合を図 1(復号化・復元) に示す．暗号化された分散情報に対して復号化・復元を行うと，排他的論理和演算の可換性から，その順序によらず一意に秘密情報が出力されることが確認できる．すなわち，秘密情報をどのような経路で暗号化および秘密分散を実行して保存しても，任意の経路で秘密情報を取得できる．

以上から，可換性を持つ演算のみで暗号処理と秘密分散処理を実現すれば，暗号処理と可換な秘密分散法が構成できることが確認できる．したがって，秘密分散法が可換性を満たすための要件は，暗号手法および秘密分散法を構成する演算が共に可換性を有することである．

2.3 秘密計算

Shamir の (k, n) 閾値法 [10] を例にとる．q を大きな素数として，2 つの秘密情報を $S_1, S_2 \in \mathbb{Z}_q$，乱数を $a_1, \cdots, a_{k-1}, b_1, \cdots, b_{k-1} \in \mathbb{Z}_q$ とする．S_1, S_2 をそれぞれ，$x = 1, \cdots, n$ において，

$$\begin{cases} f(x) = S_1 + a_1 x + \cdots + a_{k-1} x^{k-1} \mod q \\ g(x) = S_2 + b_1 x + \cdots + b_{k-1} x^{k-1} \mod q \end{cases} \tag{7}$$

のように n 個に秘密分散を行うとする．すべての x において，分散情報をそれぞれ加算すると，

$$f(x) + g(x) = (S_1 + S_2) + (a_1 + b_1)x + \cdots + (a_{k-1} + b_{k-1})x^{k-1} \mod q \tag{8}$$

という多項式が n 個得られ，これらの加算結果から和 $S_1 + S_2$ が出力可能である．実際には，n 個の分散情報は n 個の主体によってそれぞれ保持され，各主体がそれぞれローカルに $f(x) + g(x)$ の計算を行う．一方，乗算 $f(x)g(x)$ は，マルチパーティ計算で実行することで，S_1, S_2 を秘匿したまま積 $S_1 S_2$ を計算できる．しかし，その際，$f(x)g(x)$ の次元削減およびランダム化のために $n(n-1)$ 回の通信を必要とする [11].

Shamir の方式は，乗算および加算で構成されており，これらの演算が可換性を持つ算術演算であることから秘密計算が成り立つ．すなわち，秘密計算は要件として，秘密分散法が可換性を持つ算術演算で構成されることを必要とする．

3 加法に基づく $(2, n)$ 閾値秘密分散法

本節では，可換性を有し，かつ秘密計算が実現できる秘密分散方式として，加法に基づく新しい $(2, n)$ 閾値法を提案する．

3.1 要件

2.2 で確認した可換性と秘密計算の要件から，可換性を有し，かつ秘密計算可能な秘密分散法を構成するための要件は以下となる．

[要件] 暗号手法および秘密分散法を構成する演算が可換性を持つ算術演算である．

可換性を持つ算術演算として，加算演算を用いることを提案する．暗号化および秘密分散には正数の加算を用い，復号化および復元は負数の加算 (減算) に基づき実行される．

3.2 提案法の分散処理および復元処理

(a) 分散処理
$\mathcal{P}$ を n 人の管理者の集合，$\mathcal{D}$ をディーラーとする．秘密情報 S から n 個の分散情報 W_i $(0 \le i \le n-1)$ を生成する手順を示す．ただし，n が合成数である場合，素数 n_p $(> n)$ を選び，n_p 個の分散情報を生成し，その中から n 個を取り出す．

STEP 1. $\mathcal{D}$ は，秘密情報 S を $n-1$ 個の d ビットの部分秘密情報に分割する．

$$S = s_1 \| s_2 \| \cdots \| s_{n-1} \tag{9}$$

ただし，

$$s_i \in \{0,1\}^d$$
$$s_0 \in \{0\}^d$$

とする．

STEP 2. $\mathcal{D}$ は，サイズ d ビットの乱数 R_m を独立に $n-1$ 個生成する．ただし，$0 \le m \le n-2$ である．

STEP 3. 算術加算演算を実行する前に，桁上がりを考慮して s_i の上位ビットに対して 1 ビットの 0 挿入を行い，マージンを確保する．

$$s_i \leftarrow 0 \| s_i \quad (s_i \in \{0,1\}^{d+1}) \tag{10}$$

STEP 4. $\mathcal{D}$ は，部分分散情報 $w_{(i,m)}$ を生成する．

$$w_{(i,m)} = s_{i-m \bmod n} + R_m \tag{11}$$

ここで，加法の性質から $w_{(i,m)}$ は $w_{(i,m)} \in \{0,1\}^{d+1}$ である．

表 3: 加法に基づく $(2,n)$ 閾値法.

W_0	s_0+R_0	$s_{n-1}+R_1$	$\cdots$	s_2+R_{n-2}
W_1	s_1+R_0	s_0+R_1	$\cdots$	s_3+R_{n-2}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
W_{n-1}	$s_{n-1}+R_0$	$s_{n-2}+R_1$	$\cdots$	s_1+R_{n-2}

STEP 5. $\mathcal{D}$ は，$w_{(i,m)}$ を連結して表 3 のように各分散情報 W_i を生成し，管理者 $P_i \in \mathcal{P}$ に配布する．

$$W_i = w_{(i,0)} \| w_{(i,2)} \| \cdots \| w_{(i,n-2)} \tag{12}$$

(b) 復元処理

分散情報の 1 つ W_i $(0 \le i \le n-1)$ を持つ n 人の管理者 P_i の内，任意の 2 人が分散情報を共有したとき秘密情報 S が復元される．すなわち，任意の 2 人の管理者 P_i, P_j $(i \neq j)$ は互いに W_i, W_j を共有したとき，それぞれ Procedure1 に従うことで S が復元される．以下，その手順を説明をする．

STEP 1. 分散情報 W_i, W_j から部分分散情報 $w_{(i,m)}$, $w_{(j,m)}$ を取り出す.

STEP 2. $t = i - j \bmod n$ を求める.

STEP 3. 共有する 2 つの部分分散情報 $w_{(i,m)}, w_{(j,m)}$ の加減算により，$n-1$ 個の部分秘密情報 s_i $(i = 1, \cdots, n-1)$ を求める．ただし，Procedure1 において，$w_{(i,m)}, w_{(j,m)}$ の加算は $w_{(j,m)}$ の符号変換を行ってから実行され，また，$s_i < 0$ であるとき s_i の符号変換を行う.

STEP 4. STEP 3. で得られたすべての部分秘密情報を連結し，秘密情報 S を復元する.

3.3 暗号化

提案法は，従来の排他的論理和演算で構成される秘密分散法と同様に，暗号処理との可換性を有している．ただし，提案法に適用可能な暗号手法は，算術加算演算に基づく特別な乱数加算暗号の適用に限定されている．

ここで，(a)((1) 暗号化 → (2) 秘密分散) と (b)((1) 秘密分散 → (2) 暗号化) の 2 つの場合に分けて考える．

(a) (1) 暗号化 → (2) 秘密分散

まず，秘密情報 S の暗号化は，$s_i \in \{0,1\}^d$ であるため，$n-1$ 個の独立な乱数 $K_1, \cdots, K_{n-1} \in \{0,1\}^d$ を秘密鍵として，

$$C = s_1 + K_1 \| \cdots \| s_{n-1} + K_{n-1} \tag{13}$$

Procedure 1 復元

Input: W_i, W_j.

Output: S.

```
//STEP 1
Divide W_i, W_j into
(w_(i,0), ..., w_(i,n-2)), (w_(j,0), ..., w_(j,n-2)), respectively;
//STEP 2
t = i − j mod n;
//STEP 3
v = 1; s_0 = 0;
while i − vt mod n ≠ n − 1 do
    s_{vt mod n} = w_(i,i−vt mod n) + (−w_(j,i−vt mod n)) + s_((v−1)t);
    v + +;
end while
v = 1; s_0 = 0;
while j + vt mod n ≠ n − 1 do
    s_{−vt mod n} = −(w_(i,j+vt mod n) + (−w_(j,j+vt mod n)) + s_(−(v−1)t));
    v + +;
end while
//STEP 4
return S = s_1 || s_2 || ... || s_{n−1};
```

で実行される．分散処理は，先の分散処理において，S を C に，さらに W_i および $w_{(i,m)}$ は $W^C_{(i,m)}$ および $w^c_{(i,m)}$ に代えて実行される．ただし，$c_i \in \{0,1\}^{d+1}$ であるため，$R_m \in \{0,1\}^{d+1}$ となり，$w^c_{(i,m)} \in \{0,1\}^{d+2}$ となる．

(b) (1) 秘密分散 → (2) 暗号化

次に，分散情報 W_i に対する暗号化は，$w_i \in \{0,1\}^{d+1}$ であるため，$n-1$ 個の独立な乱数 $K_1, \cdots, K_{n-1} \in \{0,1\}^{d+1}$ を秘密鍵として，

$$w^c_{(i,m)} = w_{(i,m)} + K_{i-m \bmod n} \tag{14}$$

で実行される．したがって，$w^c_{(i,m)} \in \{0,1\}^{d+2}$ となる．

3.4 暗号化および秘密分散実行のためのマージン設定

暗号化および秘密分散は共に，部分情報 $(s_i, c_i, w_{(i,m)})$ に対して部分情報と同サイズの乱数を加算することで実行される．したがって，3.2 の分散処理 STEP3 のようにマージンを確保してから暗号化または秘密分散を行う必要がある．すなわち，Me 回の暗号化および階層的に Ms 回の秘密分散行う場合に最終的に必要となるマージン Ma (bit) は，

$$Ma = Me + Ms \tag{15}$$

となる．実際の処理では，3.2 の分散処理 STEP3 のように，各処理 (暗号化，秘密分散) の度に 1 ビットの 0 挿入を行い，マージンを確保してから処理を行えばよい．た

表 4: 提案法による秘密加算.

W_0	$(s_{1,0}+s_{2,0})+(R_{1,0}+R_{2,0})$	$(s_{1,n-1}+s_{2,n-1})+(R_{1,1}+R_{2,1})$	$\cdots$	$(s_{1,2}+s_{2,2})+(R_{1,n-2}+R_{2,n-2})$
W_1	$(s_{1,1}+s_{2,1})+(R_{1,0}+R_{2,0})$	$(s_{1,0}+s_{2,0})+(R_{1,1}+R_{2,1})$	$\cdots$	$(s_{1,3}+s_{2,3})+(R_{1,n-2}+R_{2,n-2})$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
W_{n-1}	$(s_{1,n-1}+s_{2,n-1})+(R_{1,0}+R_{2,0})$	$(s_{1,n-2}+s_{2,n-2})+(R_{1,1}+R_{2,1})$	$\cdots$	$(s_{1,1}+s_{2,1})+(R_{1,n-2}+R_{2,n-2})$

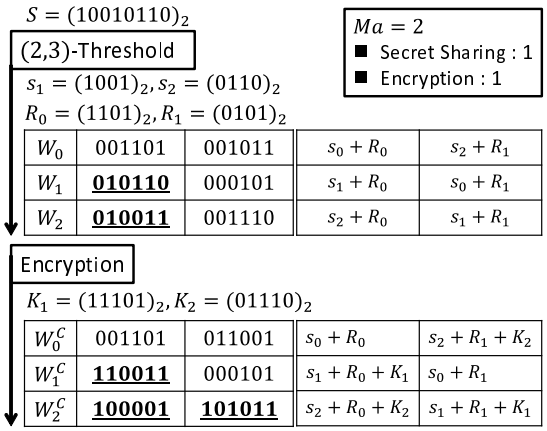


図 2: 秘密分散 1 回，暗号化 1 回のマージンの設定例 ($Ma=2$).

だし，3.2 では $Ms=1$ の設定に，3.3 の (a) および (b) では，$Me=1, Ms=1$ の設定に相当する．具体例として，図 2 に $Me=1, Ms=1$ ($Ma=2$) でマージン設定した場合を示す．ただし，暗号化は 3.3 の (b) で実行している．

4 秘密計算

4.1 概要

提案法は，算術加算演算の可換性から，秘密計算を実現可能としている．本節では，秘密情報 S_1, S_2 を秘匿したまま和 S_1+S_2 を求めることを例として，秘密加算について説明する．ただし，本稿では S_1 および S_2 が同サイズと仮定して議論される．異なる場合は，同サイズにしてから処理される．具体的には，$S_1 > S_2$ のとき，S_2 の上位ビットに $(\mathrm{MSB}(S_1)-\mathrm{MSB}(S_2))$ ビットの 0 挿入を行う．

S_1, S_2 から生成された分散情報 $W_{1,i}, W_{2,i}$ を持つ管理者 P_i がそれぞれ $W_{1,i}+W_{2,i}$ を実行すると，計算結果の部分分散情報 $w_{(i,m)}$ は以下となる．

$$\begin{aligned} w_{(i,m)} &= w_{1,(i,m)} + w_{2,(i,m)} \\ &= (s_{1,i-m \bmod n} + s_{2,i-m \bmod n}) + (R_{1,m}+R_{2,m}) \\ &\quad (w_{(i,m)} \in \{0,1\}^{d+2}) \end{aligned} \tag{16}$$

実行結果は表 4 で表される．これは表 3 と同様の構成であるため，Procedure1 に従えば，W_i, W_j から S_1+S_2 が出力されることがわかる．

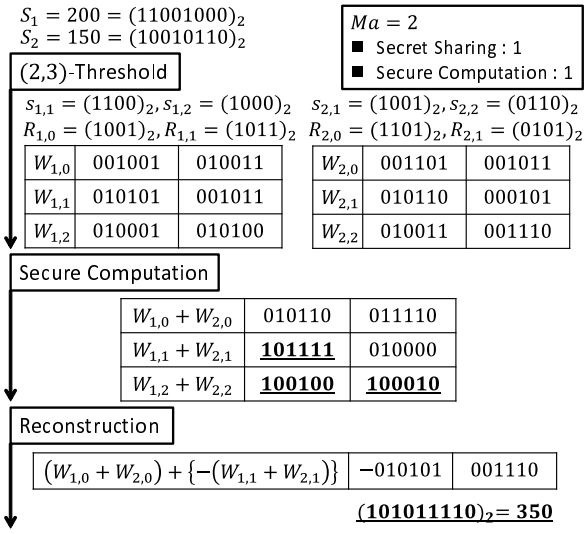


図 3: 秘密計算 1 回のマージンの設定例 ($Ma=2$).

4.2 秘密計算のためのマージン設定

3.3 では暗号化および秘密分散を併用する場合のマージンの確保の仕方について述べた．本節では，秘密加算の実行に必要なマージンを求める．秘密加算では，分散情報同士の足し算であるため，$W_{1,i}, W_{2,i}$ の MSB が同一であれば桁上がりが生じる．したがって，1 回秘密分散した分散情報を用いて Mc 回の秘密加算を行う場合，必要となるマージン Ma は，

$$Ma = 1 + Mc \tag{17}$$

となる．実際には，3.2 の分散処理 STEP3 のように，秘密加算毎に 1 ビットの 0 挿入を行い，マージンを確保してから実行する．最終的には Ma ビットのマージンが必要となる．具体例として，図 3 に $Ms=1, Mc=1$, ($Ma=2$) でマージン設定した場合を示す．

4.3 マージン設定の独立性

3.3 および 4.2 では暗号化と秘密計算とに分けてマージンの設定法について述べた．提案法は，可換な算術加算演算のみで構成されるため，それによって構成される

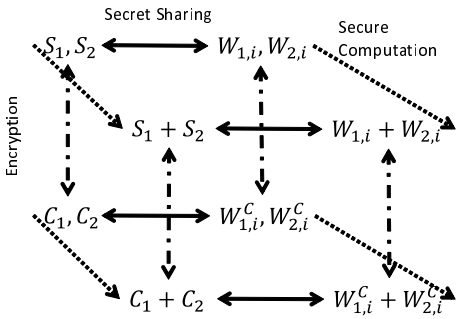


図 4: 暗号化・秘密分散・秘密計算の間の可換性.

処理の間にはすべて可換性を有する．したがって，秘密分散・暗号化・秘密計算は，任意のタイミングでそれぞれ実行される．ただし，最終的なマージン Ma は

$$Ma = Me + Ms + Mc \tag{18}$$

であり，部分分散情報 $w'_{(i,m)}$ は $w'_{(i,m)} \in \{0,1\}^{d+Ma}$ となる．

4.4 暗号化・秘密分散・秘密計算の間の可換性

文献 [12] では，暗号化した秘密情報に対して秘密分散を実行し，秘密情報を取得するときには，復元処理を行ってから復号化を行う手法が示された．これにより，秘密情報の分散処理，分散情報の分配およびストレージ，秘密情報の復元処理のすべてを，暗号化領域で実行することが可能となるため，それらを保護できるとしている．暗号処理と可換な秘密分散法であれば上の性質は有する (ただし，[12] とは，公開鍵暗号である Paillier 暗号 [13] を用いて実現されている点で異なる．)．さらに秘密計算処理と可換であれば，秘密計算も暗号化領域で実行可能であり，秘密計算処理も含めてすべての処理が暗号化領域で実現可能となる．各処理の間の可換性から，実際に，暗号化された分散情報 $W^C_{1,i}, W^C_{2,i}$ を用いて秘密加算を実行すると以下のようになる．

$$\begin{aligned} w^C_{(i,m)} &= w^C_{1,(i,m)} + w^C_{2,(i,m)} \\ &= (s_{1,i-m \bmod n} + s_{2,i-m \bmod n}) + (R_{1,m} + R_{2,m}) + \\ &\qquad (K_{1,i-m \bmod n} + K_{2,i-m \bmod n}) \end{aligned} \tag{19}$$

ここで乱数 $(R_{1,m} + R_{2,m})$ は Procedure1 に従って除去することができ，乱数 $(K_{1,i-m \bmod n} + K_{2,i-m \bmod n})$ は秘密鍵 $(K_{1,i-m \bmod n} + K_{2,i-m \bmod n})$ を用いて除去することができる．以上から，暗号処理を併用不可な従来の秘密分散法と比較して，提案法では安全性の向上が期待される．暗号化・秘密分散・秘密計算の間の可換性の様子を図 4 に示す.

4.5 秘密計算による桁上がり

Procedure1 の STEP 4 では，秘密計算結果 $s_{1,i} + s_{2,i}$ の桁上がりを考慮していない．秘密情報の桁上がりを考慮

Procedure 2 STEP 4'

Input: $s_1, s_2, \cdots, s_{n-1}$.

Output: S.

```
for t = n − 1 to 2 do
  for u = 1 to Mc do
    (s_{t−1})_u ← (s_{t−1})_u + (s_t)_{u+d};
  end for
end for
return S = s_1 ‖ s_2 ‖ ··· ‖ s_{n−1};
```

した STEP 4' は，Procedure2 に示される．ただし，$(s_t)_u$ は s_t の u ビット目を表す．なお，Procedure2 の加算処理は，桁上がりを考慮して実行される．

5 効率

5.1 情報比

一般に，秘密分散法の効率は，以下の情報比

$$\rho = \frac{\log_2 |S|}{\max_{P_i \in \mathcal{P}} \log_2 |W_i|} \tag{20}$$

で評価される [14]．提案法の情報比は，式 (20) から以下で表される．

$$\rho = \frac{(n-1) \times d}{(n-1) \times (d + Ma)} = \frac{d}{d + Ma}. \tag{21}$$

$Ma \geq 1$ であるため，他の秘密計算可能な秘密分散法と同様に，提案法は $\rho = 1$ の理想的な秘密分散法ではない．秘密計算を実現する場合，一般的に $\rho = 1$ は成り立たない．例えば，Shamir の (k, n) 閾値法では，ビット幅 α の秘密情報同士の秘密加算を行う場合，法 q の設定を 2.3 の条件かつ q $(> 2\alpha)$ として秘密分散を行うため，$\rho = 1$ とはなり得ない.

5.2 加算演算回数

提案法は，Procedure1-STEP 3 において，平均 $2\{|S| + (n-1)(Ms + Me)\}(1 - \frac{1}{n})$ ビットの，Procedure2 において，高々 $(n-2)\sum_{i=0}^{Mc-1}(d + Mc - i)$ ビットの算術加算演算を必要とする．一方，[2] では，平均 $2|S|(1 - \frac{1}{n})$ ビットの排他的論理和演算で S を復元できる．提案法と [2] との差は，演算回数だけで比較すると，$2(n-1)(Ms+Me)+(n-2)\sum_{i=0}^{Mc-1}(d + Mc - i)$ 回である．

しかしながら，排他的論理和演算による秘密分散法では，秘密計算を実現することができない.

5.3 通信量

一般に，秘密計算はマルチパーティプロトコルによる協調計算で実行される．マルチパーティ計算では，通信量がボトルネックとなっており，通信量の低減のための研究が成されている．秘密加算については，従来法 [10]

も提案法も同様にローカルで実行されるため，相互通信を必要としない．秘密乗算については，従来法を用いた場合，多項式の次元上昇に伴う次元削減およびランダム化が必要となり，n 人の管理者 P_i で相互に通信を行うため，$n(n-1)$ 回の通信が必要となる [11]．一方，提案法を用いた場合，多項式を利用していないため，次元削減およびランダム化の処理が生じない．したがって，提案法を用いてマルチパーティ計算によって積を求める場合，ローカルに計算すればよいため，相互通信を必要としないことが期待される．

6 まとめ

本稿では，加法に基づく $(2, n)$ 閾値法を提案した．提案法を構成する演算は算術加算演算のみであるため，算術加算演算の可換性から，乱数加算暗号 (算術加算) による暗号処理と可換であり，さらには秘密計算も実現可能である．また，秘密計算のための加算も可換であるため，秘密分散および暗号化を併用して秘密計算を実行でき，安全性の向上が期待される．さらに，情報比，演算回数，および通信量の観点から，効率性の高い秘密計算の実現が期待される．

参考文献

[1] 須賀祐治, “クラウド環境に適した秘密分散共有法の初期検討,” SCIS2011, no.3F1-4, 2011.

[2] 藤井吉弘, 多田美奈子, 保坂範和, 栃窪孝也, 加藤岳人, “高速な (2, n) 閾値法の構成法とシステムへの応用,” CSS2005.

[3] 藤井吉弘, 栃窪孝也, 保坂範和, 多田美奈子, 加藤岳人, “排他的論理和を用いた (k, n) しきい値法の構成法, ISEC2007-05.

[4] 栗原淳, 清本晋作, 福島和英, 田中俊昭, “排他的論理和を用いた高速な (4, n) 閾値秘密分散法と (k, n) 閾値法への拡張,” ISEC2007-04.

[5] J. Kurihara, S. Kiyomoto, K. Fukushima, and T. Tanaka, “On a fast (k, n)-threshold secret sharing scheme,” IEICE Trans. Fundamentals, vol.91-A, no.9 Sep. 2008.

[6] 高荒亮, 岩村恵市, “XOR を用いた高速な (k, L, n) ランプ型秘密分散法に関する研究, CSS2009, no.B9-3, 2009.

[7] 須賀祐治, “排他的論理和を用いた (k, n) 閾値秘密分散法の新しい構成とその優位性について,” CSS2012, no.1C2-4, 2012.

[8] 千田浩司, 五十嵐大, 高橋克己, “効率的な 3 パーティ秘匿関数計算の提案とその運用モデルの考察,” 情処学技報 2010-CSEC-48(1), pp.1–7, Feb. 2010.

[9] 布川敦史, 渡辺泰平, 須賀祐治, 岩村恵市, “計算主体を限定しない汎用的で軽量な秘匿関数計算の安全性とその拡張,” SCIS2013, 4B2–1, 2013.

[10] A. Shamir,“How to share a secret,” Commun. ACM, vol.22, pp.612–613, 1979.

[11] M. Ben-Or, S. Goldwasser, and A. Wigderson, “Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computations,” Proc. Annual Symp. Theory of Computing, pp. 1–10, ACM, 1988.

[12] B. Zhao, and E. Delp, “Secret Sharing in the Encrypted Domain,” Proc. IEEE ICC 2011, Kyoto, Japan, Jun. 2011.

[13] P. Paillier, “Public-Key Cryptosystems Based on Composite Degree Residuoity Classes,” Advanded in Cryptology — EUROCRYPT 1999, pp.223–238, 1999.

[14] D.R. Stinson, “Cryptography: theory and practice,” CRC Press, 1995.