計算コストの小さい準最適な不正検知可能秘密分散法

HOSHINO, Hidetaka / 星野, 英貴

(出版者 / Publisher)
法政大学大学院情報科学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 情報科学研究科編 / 法政大学大学院紀要. 情報科学研究科編
(巻 / Volume)
12
(開始ページ / Start Page)
1
(終了ページ / End Page)
6
(発行年 / Year)
2017-03-31
(URL)
https://doi.org/10.15002/00014412

計算コストの小さい準最適な不正検知可能秘密分散法
An Almost Optimum Cheating Detectable Secret Sharing Scheme with Low Computational Cost

星野　英貴
Hidetaka Hoshino
法政大学情報科学研究科情報科学専攻
E-mail: hidetaka.hoshino.6i@stu.hosei.ac.jp

Abstract

A cheating detectable secret sharing scheme is a secret sharing scheme that can detect forged shares in reconstructing a secret. For example, if we store shares in cloud storage, there is a possibility of it being forged. If the administrators of cloud storage are malicious, it is easy for them to forge a share. Therefore, cheating detectable secret sharing schemes have attracted attention, and many efficient schemes have been proposed. However, most existing schemes are not suitable for implementation. The reasons are as follows. First, the computational cost of the schemes is very high. Second, the required finite field for implementation depends on the secret. Finally, the schemes do not support secrets that are bit strings.

In this paper, we propose a cheating detectable secret sharing scheme suitable for implementation. However, we assume that cheaters do not know the secret. The basic idea is a bit-decomposing technique. The bit length of the proposed scheme is an optimum. Moreover, the proposed scheme is applicable to any linear secret sharing schemes.

1. 研究背景

近年，クラウドを利用したストレージサービスの利用者が増加している．クラウドを利用したストレージサービスとは，ユーザがネットワーク上にデータを保存できるサービスである．クラウドストレージによってユーザはアカウント情報があれば，どんなパソコンやスマートフォンからでも情報を引き出すことが可能になり，情報が保存されている特定のパソコンやメモリーカードを持ち歩く必要が無くなる．しかしながら，このようなサービスにはサーバへの攻撃や災害などによって預けていたデータが失われる，あるいは漏洩するといった危険性が存在している．

上記のような危険を防止するセキュリティ技術の一つとして秘密分散法が知られている．秘密分散法は，秘密情報をシェアと呼ばれるいくつかの部分情報に分散し保存する技術である．秘密分散法は次の 2 つの特徴を持っている．決められた数のシェアを集めることで元の秘密を復元することができる．決められた数未満のシェアからは秘密に関する情報は 1 ビットも漏れない．このような秘密分散法を利用する事でサービスのユーザは複数のサーバなどのストレージに情報を分散して保存することが可能になる．情報を分散して保存することで決められた数未満までのシェアが流出したとしても，秘密に関する情報は 1 ビットも漏れず，いくつかのシェアが失われた場合にも決められた数のシェアが残っていれば，残ったシェアから秘密を復元できるため情報を喪失することも無くなる．したがって，秘密分散法を利用することで，不正者からの攻撃や災害に対して，より強いセキュリティ状態で情報を保存しておくことが可能となる．しかし，このような秘密分散法にはシェアを偽造，あるいは改ざんされると正しい秘密が復元されず，誤った情報が復元されてしまうという問題点が存在している．秘密の復元時に用いるシェアのうち一つでも改ざん，偽造されれば，正規の利用者は正しい秘密を復元することが出来なくなるため，非常に大きな問題となっている．

このような問題を解決する技術として不正検知可能な秘密分散法がある．不正検知可能な秘密分散法とは，シェアに不正が行われていないことを確認するための値である認証子を付加することでシェアを偽造，改ざんする不正を検知する技術である．具体的には，秘密を復元する際に秘密と認証子の間に，決まった関係が成り立っているかを調べ，関係が成り立っていれば秘密が復元される技術となっている．

不正検知可能な秘密分散法には，二つのモデルが存在している．一つ目は不正者が秘密の内容を知っている場合でも不正を検知することができるようになっているモデルである．これを CDV モデルという [3]．二つ目のモデルは，不正者が秘密の内容を知らない場合に不正を検知することできるというモデルである．このようなモデルを OKS モデルという [4]．CDV モデルでは，不正者が秘密を知っている場合にも安全性を保証できるため，不正者が秘密を知らない場合についても安全性を保証することができる．しかしながら，CDV モデルは OKS モデルに比べてシェアサイズが大きくなるなどの理由で効率が悪くなっている．したがって，不正者が秘密を知らないという仮定の下では OKS モデルのほうが効率が良くなる．

現在までに，いくつもの不正検知可能な秘密分散法の方式 [1][2][5] が提案されているが，いくつかの課題が残されている．特に，現在提案されている不正検知可能な秘密分散法の多くは，計算にかかるコストが大きいため実装に適していない．そこで本論文では，OKS モデルの下で，計算コストが小さく，十分な安全性を持つ，

Superviser: Prof. Satoshi Obana

実装に適した不正検知可能な秘密分散法の提案を行う．

2. 準備

2.1. (k, n) 閾値型秘密分散法

線形秘密分散法の一つとして，(k, n) 閾値型秘密分散法が知られている．この技術は, シャミアによって 1979 年に提案された [6]．保護したい秘密を n 個のシェアに分散する技術で次のような性質を持っている．

- n 個のシェアのうち $k(\leq n)$ 個を集めると元々の秘密を復元することができる
- $k-1$ 個までのシェアからは秘密に関する情報は 1 ビットも漏れない

(k, n) 閾値型秘密分散法を利用することで，サーバへの攻撃による情報漏洩が起きたとしても漏洩したシェアが $k-1$ 個以下であれば秘密を知られる危険性がなくなる．また，自然災害などによるサーバの喪失が起こりシェアが失われた場合にも，k 個のシェアが残っていれば秘密を復元することができる．

シャミアの (k, n) 閾値型秘密分散法の構成法を以下に示す．

- シェア生成
  p を素数とし、秘密を $s \in GF(p^m)$ とする．ランダムな値 $a_i(1 \leq i \leq k-1)$ を選び，$GF(p^m)$ 上で定義される，次のような多項式 $f(x)$ を作る．
  $$f(x) = s + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1}$$
  参加者に配られるシェア $v_i(1 \leq i \leq n)$ は次のようになる．
  $$v_i = f(i)$$

- 秘密の復元
  $k-1$ 次の方程式は k 個の点が分かると一意に定まるため，参加者の k 個のシェアから秘密 s を $s = f(0)$ により求めることができる．

2.2. 秘密分散法の問題点

前述のように秘密分散法を利用する事で，情報の漏洩や喪失を防ぐことができる．しかし，悪意を持った不正者によってシェアが改竄，又は偽造された場合に誤った情報が復元され元々の秘密を復元できなくなるという問題が存在する．加えて，このような不正行為が行われた際に参加者が元々の秘密を知らなければ，その参加者は誤った情報が復元されたことに気がつくことはできない．このような不正は，復元に使用される k 個のシェアのうちの一つを改竄するだけで実行可能となっているため，危険性が高いと言える．このような問題を解決するために，秘密の復元時にシェアに対して不正が行われていないことを確かめ，もし不正が行われていれば，その事実を参加者に知らせることができる不正検知可能な秘密分散法が提案されている．

2.3. 不正検知可能な秘密分散法

参加者に配られるシェアを，(1) 秘密の部分情報と (2) 不正を検知するための認証子となる情報の部分情報という二つの値のセットとすることで，秘密の復元時に不正の有無を確認できる．このような技術を不正検知可能な秘密分散法という．認証子となる情報の作り方で不正を検知できる確率や利用可能な体などが変化する．

■不正検知可能な秘密分散法のシェア　不正検知可能な秘密分散法のシェアの一般的構成法は以下のようになる．秘密 s から認証子となる値を生成するための関数を $A(x)$ とする．秘密 s を分散し $v_{1i}(1 \leq i < n)$ を生成する．次に，関数 $A(s)$ の出力を分散し $v_{2i}(1 \leq i < n)$ を生成する．参加者にシェアとして二つの値のセット $V_i = (v_{1i}, v_{2i})(1 \leq i < n)$ が配られる．

■不正の検知　参加者のシェアから次の二つの値が復元される．

- $\hat{s}$:秘密 s のシェアから復元された値
- $\hat{a}$:認証子のシェアから復元された値

この時，次の式が成り立つことを確認する．

$$\hat{a} = A(\hat{s})$$

不正が行われていない場合，この式は常に成り立つ．そのため，この式が成り立っている場合，不正は行われていないと判断して秘密として $\hat{s}$ を出力する．反対に，この式が成り立たない場合は不正が行われたと判断して，不正があったことを参加者に知らせるシンボル $\perp$ を出力する．この時，秘密が復元されることはない．

■シェアサイズの下界　不正者が秘密を知らない場合において秘密が一様分布の場合に不正検知可能な秘密分散法のシェアサイズの下界は次のようになる [4].

$$|V_i| \geq \frac{|S|-1}{\epsilon} + 1$$

3. 既存の方式

3.1. 秘密の二乗を利用した方式

[1] の論文で次のような不正検知可能な秘密分散法が提案されている．この方式は不正者が秘密を知らない場合モデルで $GF(p)$ 上で，通常の秘密分散法と同様に秘密 s を分散した値に加えて秘密の二乗の値 s^2 を分散した値のセットをシェアとして，参加者に配布する方式となっている．この方式における $\hat{s}$, $\hat{a}$ は次のようになる．

- $\hat{s}$:秘密 s のシェアから復元された値
- $\hat{a}$:認証子 s^2 のシェアから復元された値

秘密の復元時に，次の式が成り立っていれば不正が行われていないことがわかる．

$$\hat{a} = \hat{s}^2$$

この式が成り立たなければ，不正があったことを参加者に知らせる．

■不正成功確率と問題点　この方式の不正成功確率は $1/p$ となっている．しかしながら，$p = 2^n$ の場合には確率 1 で不正に成功してしまうという問題点を持っている．そのため，この方式は秘密が $GF(2^n)$ の要素となっている場合に利用できない方式となっている．コンピュータでよく利用されるビット列は $GF(2^n)$ の要素であるため，この方式は秘密がビット列の場合に利用できず，利便性が低くなっている．

3.2. 秘密がランダムなビット列である場合に適した方式

[2] では，秘密がよく利用されるビット列である場合に適した方式を提案している．この方式は，秘密がランダムなビット列の場合に適しているため $GF(2^n)$ 上で利用可能な方式となっている．この方式は，不正者が秘密を知らない場合に安全性を保証する方式となっている．この方式では，秘密 s と秘密の 3 乗の値 s^3 をそれぞれ分散し，シェアとして参加者に配布することで秘密が $GF(2^n)$ の時にも不正を検知する．この方式における，$\hat{s}$，$\hat{a}$ は次のようになる．

- $\hat{s}$:秘密 s のシェアから復元された値
- $\hat{a}$:認証子 s^3 のシェアから復元された値

秘密の復元時に，次の式が成り立っていれば不正が行われていないことがわかる．

$$\hat{a} = \hat{s}^3$$

この式が成り立たなければ，不正があったことを参加者に知らせる．

■不正成功確率と問題点　この方式の不正成功確率は $2/p$ となっている．秘密が l ビットのビット列の場合には，不正成功確率は $2/2^l = 1/2^{l-1}$ となる．この方式は一般化することで，任意の値 l' に対して，不正成功確率を $1/2^{l'}$ とすることができる．しかし，この方式のシェアサイズは，$l \geq l' + 1$ の場合に下界より 1 ビット長くなり，$l \leq l'$ の場合に下界より 2 ビット長くなる．したがって，この方式の問題点はシェアサイズが理論限界を満たしていない点である．また，この方式は秘密をビット列と仮定しているため，$p = 3^n$ の場合には確率 1 で不正に成功する．

3.3. ビット列の分割を用いた方式

[8] では，秘密がビット列の場合に，シェアのビット長が最適となる方式が提案されている．この方式は，秘密がランダムで，ビット長が偶数の場合に利用できる方式となっている．つまり，秘密が $GF(2^{2m})$ 上で一様に分布している場合に安全性を保障する方式となっている．また，この方式も OKS モデルで安全性を保障する方式となっている．この方式の，不正の検知を行うために，秘密のビット列を半分に分割し s_1 と s_2 を生成し，それぞれを掛け合わせた値 $s_1 \cdot s_2 \in GF(2^m)$ を利用している．したがって，この方式における，$\hat{s}$，$\hat{a}$ は次のようになる．

- $\hat{s}$:秘密 s のシェアから復元された値
- $\hat{a}$:認証子 $s_1 \cdot s_2$ のシェアから復元された値

秘密の復元時に，次の式が成り立っていれば不正が行われていないことがわかる．

$$\hat{a} = \hat{s_1} \cdot \hat{s_2}$$

この式が成り立たなければ，不正があったことを参加者に知らせる．

■不正成功確率と問題点　この方式に対する不正成功確率は，$1/2^m$ となっている．しかしながら，この方式は計算量が多く，実装を行った場合に，大きい秘密に対して，現実的な時間で処理を終えることが出来ないという課題を持っている．

3.4. 計算量の小さい方式

[7] では，秘密のビット列をいくつものビット列に分割することで，計算量を小さくする方式が提案されている．この方式は，p を素数として，秘密が $(\mathbb{Z}_p^*)^N$ の要素である場合に安全性を保障できる．この方式では，分割することで生成したビット列を全て掛け合わせることで，不正の検知を行う．ビット列の分割により，小さな体で計算を行うことが出来るため，計算コストが小さくなっている．この方式における，$\hat{s}$，$\hat{a}$ は次のようになる．

- $\hat{s}$:秘密 s のシェアから復元された値
- $\hat{a}$:認証子 $s_1 \times s_2 \times \cdots \times s_N$ のシェアから復元された値

秘密の復元時に，次の式が成り立っていれば不正が行われていないことがわかる．

$$\hat{a} = \hat{s_1} \times \hat{s_2} \times \cdots \times \hat{s_N}$$

この式が成り立たなければ，不正があったことを参加者に知らせる．

■不正成功確率と問題点　この方式に対する不正成功確率は，$1/(p-1)$ となる．この方式は，$(\mathbb{Z}_p^*)^N$ の要素となっている秘密のみをサポートしているため，ビット列をサポートしていないことが課題となっている．

4. 提案方式

本節では，尾形，荒木の方式 [2] と同様に秘密がビット列であることを前提とし，不正者が秘密を知らないとする OKS モデルの下で，安全性を保証できる不正検知可能な秘密分散法の方式を提案する．加えて，本提案方式は計算量が小さく，シェアのビット長についても理論限界を満たす方式となっている．本方式では，秘密のビット列の分割を利用した認証子生成関数を用いて不正検知可能な秘密分散法を構成する．

上記の既存方式は，ビット列に対して安全を保障できない，認証子の導出に関する計算コストが非常に大きいなどの課題を持ち，実装を行う際に非常に大きな課題となっている．そこで，本方式は，実装の容易さ．ビット列のサポート，認証子の導出に関する計算コストなどの

面で，既存方式よりも効率の良い不正検知可能な秘密分散法となっている．例えば，$GF(2^{1024})$ などの大きな体の要素である秘密に対して，既存の方式を利用すると，現実的な時間で処理を終えることが出来ない．これは，大きな体上での乗算は計算コストが非常に大きくなるためである．本方式では，ビット列の分割を用いて，小さな体上で認証子の導出を行っているため，既存方式に比べて，計算コストの小さな方式となっている．

はじめに，秘密のビット列 $s \in GF(2^{mN})$ を N 個に分割することで，N 個のビット列 $s = (s_1, s_2, \cdots, s_N) \in GF(2^m)^N$ を生成する．この下で，シェアとして (1) 秘密のシェア，(2)$s_1 \cdot s_2 + s_3 \cdot s_4 + \cdots + s_{N-1} \cdot s_N$ のシェアをユーザに持たせる．以上により，不正成功確率が $1/2^m$ となる秘密分散法が構成される．この認証子 $s_1 \cdot s_2 + s_3 \cdot s_4 + \cdots + s_{N-1} \cdot s_N$ の導出を，$GF(2^{mN})$ 上ではなく，$GF(2^m)$ 上で行うことで，認証子の導出に関する計算コストを小さくすることが出来る．提案方式の詳細を以下に示す．

■秘密の分散　秘密 s を分散した値を v_{1i} とする．

$$V_1 = (v_{11}, v_{12}, \cdots, v_{1N})$$

次に，$s = (s_1, s_2, \cdots, s_N)$ に対し，$s_1 \cdot s_2 + \cdots + s_{N-1} \cdot s_N \in GF(2^m)$ を分散した値を v_{2i} とする．

$$V_2 = (v_{21}, v_{22}, \cdots, v_{2N})$$

参加者 i には，二つの値のセット (v_{1i}, v_{2i}) がシェアとして配られる．

■秘密の復元　参加者のシェア V_1, V_2 の部分集合から値を復元する．V_1 の部分集合から復元される値を $\hat{s} = (\hat{s_1}, \hat{s_2}, \cdots, \hat{s_N})$，$V_2$ の部分集合から復元される値を $\hat{a}$ とし，次のように，出力を行う..

$$\begin{cases} \hat{s} \;\; if \;\; \hat{s_1} \cdot \hat{s_2} + \cdots + s_{\hat{N-1}} \cdot \hat{s_N} = \hat{a} \\ \perp \;\; otherwise \end{cases}$$

ここで，$\perp$ は不正が起きたことを意味する出力である．したがって，上式が成り立っている場合は，不正は起きていないと判断し，式が成り立たない場合は不正が起きたと判断している．

■不正成功確率と安全性　提案方式の不正成功確率とシェアサイズについて以下の定理が成り立つ．

定理 1 $s \in S$ を $GF(2^{mN})$ で一様に分布する秘密とし，不正者は秘密を知らないものとする．このとき，任意の素数べき 2 に対し，提案方式は $\epsilon = 1/2^m$ を満たし，シェアサイズ $|V_i|$ は $|V_i| = |S|/\epsilon = 2^{mN+m}$ となる．また，本方式は任意の線形秘密分散法に適用可能である．

証明 1 不正が行われた場合，次の二つの値が復元される．

- V_1 から復元される値*:*

$$\hat{s} = (\hat{s_1}, \hat{s_2}) = (s_1 + \delta_1, s_2 + \delta_2, \cdots, s_N + \delta_N)$$

- V_2 から復元される値*:*

$$\hat{a} = s_1 \cdot s_2 + \cdots + s_{N-1} \cdot s_N + \delta_a$$

ここで，δ_1，δ_2，$\cdots$，δ_N，δ_a は不正者が不正を行ったことにより生じる元の値との差分であり，全ての線形秘密分散において不正者が任意に操作できる値である．ここで，δ_1，δ_2，$\cdots$，δ_N の全ての値を 0 にした場合，風限される秘密の値は変化しない．したがって，不正は行われていないことになるため，不正が行われた場合には δ_1，δ_2，$\cdots$，δ_N の中に少なくとも一つは 0 でない δ_i が存在する．このとき，次式が成り立てば不正者は不正に成功する．

$$\hat{a} = \hat{s_1} \cdot \hat{s_2} + \cdots + s_{\hat{N-1}} \cdot \hat{s_N}$$

$$\sum_{i=1}^{N/2} (\delta_{2i} s_{2i-1} + \delta_{2i-1} s_{2i} + \delta_{2i-1}\delta_{2i}) - \delta_a = 0$$

ここで，$\delta_i (1 \le i \le N)$ の中には，少なくとも一つ 0 でない δ_i が存在している．したがって，上式は線形方程式となっていることがわかる．また，秘密は一様に分布している．以上から，上式が成り立つ確率は $1/2^m$ となる．

加えて，不正者は δ_1，δ_2，δ_a の値を任意に操作出来るが，不正者は秘密を知らないため上式を常に成り立たせることはできない．　□

■シェアサイズ　秘密を $2mN$ ビットのビット列したとき，$p = 2^{mN}$ となる．この場合，本方式のシェアサイズは次のようになる．

$$|V_i| = 2^{mN+m}$$
$$\log |V_i| = mN + m$$

また，不正者が秘密を知らず，秘密が一様分布する場合の不正検知可能な秘密分散法のシェアサイズの下界は次のようになる．

$$\log |V_i| \ge \log \left(\frac{|S| - 1}{\epsilon} + 1 \right)$$

本方式のパラメータは $|S| = 2^{mN}, \epsilon = \frac{1}{2^m}$ となり，この方式に対するシェアサイズの下界は次のようになる．

$$\begin{aligned} \lceil \log |\mathcal{V}_{\mathsf{OKS}}| \rceil &= \left\lceil \log \left(\frac{2^{mN} - 1}{2^{-m}} + 1 \right) \right\rceil \\ &= \left\lceil \log \left(2^{mN+m} - 2^m + 1 \right) \right\rceil \\ &= \left\lceil \log \left(2^{mN+m} \left(1 - \frac{2^m}{2^{mN+m}} + \frac{1}{2^{mN+m}} \right) \right) \right\rceil \\ &= \left\lceil \log \left(2^{mN+m} \left(1 - \frac{1}{2^{mN}} + \frac{1}{2^{mN+m}} \right) \right) \right\rceil \\ &= \left\lceil \log 2^{mN+m} + \log \left(1 - \frac{1}{2^{mN}} + \frac{1}{2^{mN+m}} \right) \right\rceil \\ &= \left\lceil mN + m + \log \left(1 - \frac{1}{2^{mN}} + \frac{1}{2^{mN+n}} \right) \right\rceil \\ &= mN + m \end{aligned}$$

ここで，次の式が成り立つ．

$$-1 < \log\left(1 - \frac{1}{2^{mN}} + \frac{1}{2^{mN+m}}\right) < 0\,.$$

以上から，本方式はシェアサイズの下界をほとんど満たした方式と言える．

5. 比較

本節では，提案方式と既存方式の比較を行う．表 1の方式は，全て OKS モデルで安全性を保障する方式である．

まず，秘密をいくつかのビット列に分割してから，それぞれの分割されたビット列に対して既存の方式を用いる方法について考える．例えば，1024 ビットの秘密を 8 個の 128 ビットのビット列に分割し，8 個のビット列それぞれに既存の方式を適用する．したがって，秘密のシェアと認証子のシェアがそれぞれ 8 個生成される．また，生成されるシェアのサイズはそれぞれ 128 ビットである．このとき，8 個のシェアの組のうち 1 個の組に対して不正を行えば，復元される秘密は誤った秘密となる．したがって，不正者は 8 個のシェアの組のうち 1 組にのみ不正を行う．また，もし不正者が 2 組以上のペアに対して不正を行った場合，不正に成功する確率は低くなる．これは，不正者が 8 組すべてに対して不正を行ったとすると，8 組すべてで不正に成功しなければならないからである．なぜなら，1 組でも不正が起きたことを検知すれば，$\perp$ を出力しユーザに不正が起きたことを知らせるからである．実際にこの方法を尾形らの方式に適用した場合，次のようなパラメータとなる．

$$|\mathcal{S}| = 2^{1024}, \quad \epsilon = 2/2^{128}, \quad |\mathcal{V}_i| = 2^{2048}\,.$$

一方で，提案方式を分割数 8 で，1024 ビットの秘密に対して利用した場合のパラメータは次のようになる．

$$|\mathcal{S}| = 2^{1024}, \quad \epsilon = 1/2^{128}, \quad |\mathcal{V}_i| = 2^{1152}\,.$$

提案方式のシェアのビット長は下界を満たしている．この二つの方式のパラメータを比較すると，不正成功確率は非常に近い確率であるにも関わらず，提案方式の方がシェアのビット長がはるかに小さいことがわかる．シェアはユーザが所持するデータであるため，ビット長が大きければ大きいほどユーザの負担となる．したがって，秘密分散法を利用するうえでシェアのサイズは非常に重要なパラメータであるといえる．このように，単純な方法で，大きな秘密に対して既存方式を利用すると，非常に効率が悪く，ユーザ負担の増大につながることがわかる．また，尾形らの方式以外の既存方式についても同様に効率が悪くなる．加えて，分割数が多くなった場合や秘密のサイズが大きくなった場合には，効率はより悪くなる．したがって，この単純な方法は非常に効率が悪いといえる．以上のような利用から，本節では，既存方式をそのまま利用した場合と提案方式の比較を行う．

はじめに，実装に必要な体についての比較を行う．Cabello らの方式と尾形らの方式を実装する際には，秘密が分布している体を必要とするため，実装しなければならない体の数が非常に多くなる．例えば，秘密が $GF(2^{128})$ の要素であるときは，$GF(2^{128})$ を実装する必要がある．同様に，秘密が $GF(2^{256})$ の要素であるときは，$GF(2^{256})$ を実装する必要がある．加えて，星野らの方式もほぼ同様の性質を持っている．また，荒木らの方式の実装時には，素数 p に応じた体の実装が必要になる．一方で，提案方式を実装する場合，$GF(2^{128})$ を実装することで，$GF(2^{256})$ の要素である秘密にも適用可能となり，既存方式に比べ，実装する体の数が少なくなることがわかる．

次に，認証子の導出に関する計算量の比較を行う．計算量の比較として，各方式の乗算回数を比較する．これは，体上の加算は計算コストが乗算の比べて非常に小さく，ほとんど無視できるためである．ここで，1024 ビットの秘密に対して，求められる安全性が不正成功確率 $\epsilon = 1/2^{128}$ の場合を考える．Cabello らの方式を利用した場合，1 回の乗算を $GF(2^{1024})$ 上で行うことになる．同様に，尾形らの方式を利用した場合，2 回の乗算を $GF(2^{1024})$ 上で行うことになる．また，星野らの方式を利用した場合，1 回の乗算を $GF(2^{512})$ 上で行うことになる．一方，提案方式を利用した場合，4 回の乗算を $GF(2^{128})$ 上で行うことになる．計算コストを比較するために，$GF(2^L)$ 上の乗算の計算コストは $GF(L^2)$ になるという仮定を用いると，提案方式における認証子の導出は，尾形らの方式に比べて 32 倍早く，星野らの方式に比べて 4 倍早いことがわかる．加えて，秘密のサイズが大きくなればなるほど提案方式と既存方式の計算コストの差は大きくなっていく．例えば，秘密のサイズが $1GB$ の場合，尾形らの方式では，2 回の乗算を $GF(2^{2^{30}})$ 上で行うことになる．同様に，星野らの方式を利用した場合，1 回の乗算を $GF(2^{2^{15}})$ 上で行うことになる．実際には，このように大きな体では，計算を現実的な時間で終えることはできないため，大きな秘密に対して既存方式を利用することは難しい．対して，提案方式を利用した場合は，2^{25} 回の乗算を $GF(2^{128})$ 上で行うことになる．このように，大きな秘密に対して利用する場合も，乗算を行う回数は増加するものの計算を行う体は $GF(2^{128})$ のように比較的小さな体となっている．したがって，大きな秘密に対して，既存方式に比べ圧倒的に小さな計算コストで方式を適用することが出来る．以上から，提案方式は既存方式に比べて，計算コストの面で優れていて，大きな秘密にも適用可能な方式となっていることがわかる．

表 2　ビット列をサポートする方式の計算量

方式	乗算回数	演算を行う体	計算量
提案方式	4	$GF(2^{128})$	$4M$
尾形ら [2]	2	$GF(2^{1024})$	$128M$
星野ら [8]	1	$GF(2^{512})$	$16M$

$s \in GF(2^{1024})$, $\epsilon \leq 1/2^{128}$
M は $GF(2^{128})$ 上での 1 回の乗算コスト

表 1　方式の性質

方式	シェアサイズ	サポートする秘密	実装に必要な体，群
提案方式	$\|\mathcal{S}\|/\epsilon$ (optimum)	$GF(2^{mN})$	$GF(2^m)$
Cabello ら [1]	$\|\mathcal{S}\|/\epsilon$ (optimum)	$GF(2^m)$ 以外の任意の体	秘密が分布している体
尾形ら [2]	$2\|\mathcal{S}\|/\epsilon$	$GF(3^m)$ 以外の任意の体	秘密が分布している体
星野ら [8]	$\|\mathcal{S}\|/\epsilon(optimum)$	$GF(2^{2m})$	$GF(2^m)$
荒木ら [7]	$p^{N+1} \approx \|\mathcal{S}\|/\epsilon$	$(\mathbb{Z}_p^*)^N$	$GF(p)$

次に，サポートする秘密について比較する．まず，提案方式は $GF(2^{mN})$ の要素である秘密をサポートしている．言い換えると，mN ビットのビット列に対して安全性を保障できる方式となっている．次に，星野らの方式は $GF(2^{2m})$ の要素である秘密に対して安全性を報奨している．これは，提案方式において $N = 2$ の場合であるため，提案方式に比べサポートする秘密の制限が強いことがわかる．一方で，尾形らの方式は $GF(3^m)$ の要素である秘密以外をサポートしている．したがって，ビット列については一切の制限がなく，任意のビット列に対して安全性を保障している．また，Cabello らの方式と荒木らの方式についてはビット列をサポートしていない．

まとめると，提案方式は次のようなメリットを持っている．一つ目は，実装する体の数が少なく，実装に適していることである．二つ目は，ビット列をサポートしていることである．最後に，認証子の導出にかかるコストが小さく，大きな秘密にも適用可能であること．以上から，提案方式は，既存方式に比べて実装に適しているといえる．

6. まとめと今後の課題

本論文では，不正者が秘密を知らないという仮定の上で，秘密が $GF(2^{mN})$ 上で一様に分布する場合に不正成功確率が $1/2^m$ となる方式の提案を行った．加えて，本方式のシェアサイズは以下のプロパティを持ち理論限界をほとんど満たしている．

$$|\mathcal{S}| = 2^{mN}, \quad \epsilon = 1/2^m, \quad |\mathcal{V}_i| = 2^{mN+m}\,.$$

また，本方式は認証子の導出を $GF(2^m)$ 上で行うことで，認証子の導出にかかる計算コストが小さく，実装した際に現実的な時間で利用できる方式となっている．本方式のパラメータである m と N は，それぞれ任意の整数と任意の偶数で，認証子の導出を行う体と秘密のビット列の分割数に対応している．したがって，提案方式を用いた場合，$GF(2^m)$ 上で乗算を $N/2$ 回行うことになる．このように，大きな秘密に対しても分割数を多くすることで $GF(2^m)$ 上での演算で認証子の計算が可能になるため，計算コストを小さく抑えることが出来る．

本方式は，秘密が $GF(2^{mN})$ の要素となっている必要があり，尾形らの方式のように任意のビット列に対して，適用することが出来ない．そのため，提案方式では適用できる秘密が限られている点が課題となっている．したがって，任意の秘密に対して安全性を保障でき，実装に適した方式を構成することが今後の課題となる．

文　献

[1] Sergio Cabello, Carles Padró, Germán Sáez, “ Secret Sharing Schemes with Detection of Cheaters for a General Access Structure ” Designs, Codes and Cryptography, 25, 175-188, 2002.

[2] Wakaha OGATA, Toshinori ARAKI, “ Cheating Detectable Secret Sharing Schemes for Random Bit Strings ” IEICE TRANS. FUNDAMENTALS, VOL.E96-A, NO.11, NOVEMBER 2013.

[3] M. Carpentieri, A. De Santis and U. Vaccaro, ”Size of Shares and Probability of Cheating in Threshold Schemes,” Proc. Eurocrypt’93, Lecture Notes in Computer Science, vol. 765, Springer Verlag, pp.118-125, 1995.

[4] W. Ogata, K. Kurosawa and D. R. Stinson, ”Optimum Secret Sharing Scheme Secure against Cheating,” SIAM Journal onDiscrete Mathematics, vol. 20, no. 1, pp. 79-95, 2006.

[5] Satoshi Obana and Kazuya Tsuchida, ”Cheating Detectable Secret Sharing Schemes Supporting an Arbitrary Finite Field,” Advances in Information and Computer Security, Lecture Notes in Computer Science Volume 8639, 2014, pp 88-97

[6] A. Shamir,“ How to Share a Secret, ”Communications of the ACM, vol. 22, no. 11, pp. 612-613, 1979.

[7] T. Araki and W. Ogata, “A Simple and Efficient Secret Sharing Scheme Secure against Cheating,” IEICE Trans. Fundamentals, vol. E94-A, no. 6, June 2011, pp. 1338–1345, 2011.

[8] H. Hoshino and S. Obana, “Almost Optimum Secret Sharing Schemes with Cheating Detection for Random Bit Strings,” Advances in Information and Computer Security, Lecture Notes in Computer Science vol. 9241, Springer Verlag, pp. 213-222, 2015.