秘密分散法とその応用について

土井 洋 †

概 要

秘密分散法は秘密を分散管理するための方法として Blakley と Shamir により，1979 年に独立に提案された．現在も，秘密分散法とその周辺に関する研究は盛んになされており，守秘，認証はもちろん，複雑な暗号プロトコルを構成するためにも広く使われている．本論文では，まず，秘密分散法の例を (k, n) 閾値法を中心に紹介する．秘密分散法の応用は多岐に渡るが，本論文では守秘に関する応用例を中心に紹介する．

1 はじめに

暗号技術は，守秘と認証に大別することができる．現代では，これらを実現するためのアルゴリズムは公開されていることが多く，秘密の情報 (以下，秘密) を有するか否かで，可能な処理が異なる．例えば，公開鍵暗号では，暗号化を行う側は秘密を持たないが，暗号文を復号する側は復号するための秘密鍵を有する．認証技術の一つであるデジタル署名では，署名を生成する側が秘密を有し，署名を検証する側は秘密を有しない．

現代暗号では，アルゴリズムが公開されているので，守秘や認証を実現する方式の安全性を議論する場合は，秘密の管理が適切に行われていることを前提とする．例えば守秘の場合,「秘密鍵を有しなければ暗号文から平文の情報を得ることができない」などの性質が求められる．当然ながら，秘密鍵を紛失した場合は，暗号文を復号できなくなる．逆に，秘密鍵を何らかの手段で得たら，本来の秘密鍵の保有者と全く同じ処理 (復号処理など) を行うことができる．

秘密を分散管理し，必要な時に復元する方法は，Blakley[2] と Shamir[12] により，1979 年に独立に提案されている．そして，誕生から 30 年を経た現在も，様々な角度から研究がなされている．

本論文では，まず，秘密分散法の代表例として Shamir の (k, n) 閾値法を中心に紹介する．[2, 12] では，秘密鍵の分散管理の実現を志向していた秘密分散法であるが，後に能力 (復号能力，署名能力) の分散管理にも適用されている．そこで，復号能力の分散管理の考え方や属性ベース暗号，秘匿関数計算についても触れ，これらと秘密分散法の関連について紹介する．

†情報セキュリティ研究科 教授

2　秘密分散の考え方

秘密の保有者を A とし，A が保有する秘密を s とする．A にとって，秘密の紛失 (忘却を含む) や秘密の漏えいを防ぎたいというのはごく自然の要求である．

まず，秘密の紛失への対策について考える．例えば，秘密の保有者を更に 2 人 (これを B, C とする) 増やし，A, B および C が各々同じ秘密 s を保有することとする．すると，A, B, C のうちの 2 人が秘密 s を紛失 (または忘却) したとしても，残る 1 人の秘密が失われなければ，結果として秘密 s は失われないことになる．従って，この方法は秘密の紛失への対策になっている．ところが，この方法は秘密 s の漏えいを防ぎたいというもう一方の要求と相容れない．例えば，与えられたパスワードを忘れないように，手帳や付箋紙等に記録したことのある人は少なくないだろう．当然ながら，手帳や付箋紙が第三者の手に渡ることはパスワードの漏洩に直結する[1]．

次に，秘密の漏洩への対策について考える．秘密の保有者を A, B, C とし，秘密 s を何らかの方法で，s_A, s_B, s_C と分割したとする．ここで，s_A, s_B, s_C の 3 つが揃わない限り，秘密 s を復元できないように分割できたとしよう．そして A, B, C が各々 s_A, s_B, s_C のみを保有することとする．この場合，例えば秘密 s_A, s_B が漏洩した場合も，残る秘密 s_C が漏えいしない限り，秘密 s は漏洩しないとみなしてよい．ところが，この方法は秘密 s の紛失を防ぎたいというもう一方の要求と相容れない．A, B, C のいずれかが分割された秘密を紛失 (つまり，s_A, s_B, s_C のいずれかが紛失) した時点で，秘密 s を復元できなくなるからである．

このような秘密の漏えいと秘密の紛失への対策の一つが秘密分散法であり，3.1 節で述べる (k, n) 閾値法などが広く知られている．

3　秘密分散法

秘密分散法の基本的な考え方と簡単な例を紹介する．基本的な考え方は，連立 1 次方程式の解を求めること，もしくは行列の計算などの知識があれば理解できる．

3.1　Shamir の (k, n) 閾値法

Shamir の (k, n) 閾値法 [12] の考え方を紹介する．話を簡単にするために，秘密 s を整数とする．(k, n) 閾値法では，まず，秘密 s の保有者 (以下，ディーラ) が，s から n 個のシェア $(s_1, \cdots, s_n)$ と呼ばれる値を生成する．次に，ディーラはシェア保有者 $U_1, \cdots, U_n$ に各々シェア $(s_1, \cdots, s_n)$ を秘密裏に渡す．ディーラはこの直後に消滅してしまってもよい．その後，任意の k 人のシェア保有者が協力して k 個のシェアを集め，所定の計算をすることにより秘密 s を復元できる．k を閾値と呼ぶ．以下，シェア保有者には，各々識別子として整数 $1, \cdots, n$ が割り当てられているとする．

分散　定数項を s とするランダムな $k-1$ 次多項式

$$f(x) = a_{k-1}x^{k-1} + \cdots + a_0$$

[1] 手帳や付箋紙等に記録することは推奨されていない．

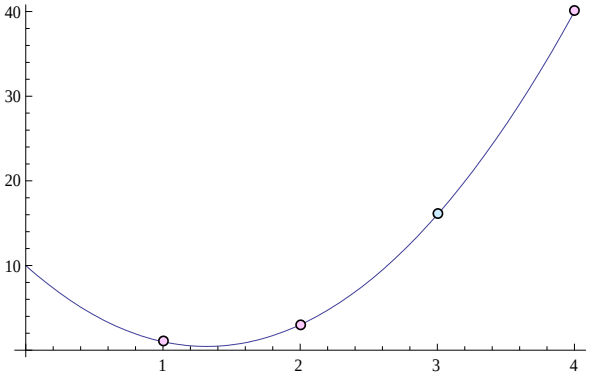

図 1: 閾値法の考え方

を生成する．ここで，$a_{k-1}, \cdots, a_0$ はいずれも整数であり，特に $a_0 = s$ である．この多項式 (係数 $a_{k-1}, \cdots, a_0$) はディーラしか知らない．$f(i)$ を U_i のシェア s_i とする．ディーラはシェア保有者 U_i にシェア $f(i)$ を秘密裏に渡す.

復元 k 人のシェア保有者が $(i, f(i))$ を持ちより，これらから $a_0 = s$ を求める．

図 1 に，$(3, 4)$ 閾値法の考え方を示す．シェア保有者に識別子 $\{1, 2, 3, 4\}$ が割り当てられ，U_1, U_2, U_3, U_4 に，$f(1), f(2), f(3), f(4)$ がシェアとして与えられているとする．ここで，秘密の多項式 f の係数 a_2, a_1, a_0 は

$$y = a_2 x^2 + a_1 x + a_0$$

上の 3 点が与えられれば定まることから，3 個のシェアが揃えば，秘密 $s(= a_0)$ を得られることが，直感的に理解できる．また，例えば s_3 が失われたとしても，残る $\{s_1, s_2, s_4\}$ を用いて秘密 s を復元できる．

秘密 s の復元が可能であることは，式を用いて以下のように確認できる．復元に協力する k 人のシェア保有者の識別子を $\{i_1, \cdots, i_k\}$ とする．まず,

$$\begin{aligned} f(i_1) &= a_{k-1} i_1^{k-1} + a_1 i_1 \cdots + a_0 \\ &\cdots \\ f(i_k) &= a_{k-1} i_k^{k-1} + a_1 i_k \cdots + a_0 \end{aligned}$$

が成り立つ．ここで，$(i_1, f(i_1)), \cdots, (i_k, f(i_k))$ が与えられれば，$i_1^{k-1}, \cdots, i_1, \cdots, i_k^{k-1}, \cdots, i_k$ も既知であるので，未知変数を $a_{k-1}, \cdots, a_0$ の k 個とする k 変数 1 次式が，k 個与えられることになる．この連立式かから，$a_{k-1}, \cdots, a_0$ を求めることができるので，秘密 s を復元できることになる．

さて，ラグランジュ補間を用いると，

1. $|S| = k$ となる集合 $S = \{i_1, \cdots, i_k\} \subset \{1, 2, \cdots, n\}$ を選び,

2. $\lambda_{i_l,S}(x) = \prod_{i_j \in S \setminus i_l} \frac{x-i_j}{i_l-i_j}$ とおくと，$f(x) = \sum_{i_l \in S} f(i_l)\lambda_{i_l,S}(x)$

となることが知られている．この特殊な形，すなわち

$$\begin{aligned} f(0) &= \sum_{i_l \in S} f(i_l)\lambda_{i_l,S}(0) \\ &= \sum_{i_l \in S} f(i_l) \prod_{i_j \in S \setminus i_l} \frac{-i_j}{i_l - i_j} \end{aligned}$$

を計算すると定数項 a_0(すなわち，秘密 s) を求めることができる．なお，集合 S は，秘密を復元するために協力するシェア保有者の識別子の集合である．

さて，ここまで秘密 s や，秘密の多項式の係数 $a_{k-1}, \cdots, a_0$ を整数として考え方を説明してきた．しかし，計算機が扱うことができる値は有限なものに限られるから，q を素数とする有限素体 $\mathbb{Z}/q\mathbb{Z}$ 上で (k,n) 閾値法が実現される場合が多い[2]．この場合，秘密の多項式の係数 $a_{k-1}, \cdots, a_0$ は $\mathbb{Z}/q\mathbb{Z}$ 上の元である (秘密 $s = a_0$ も $\mathbb{Z}/q\mathbb{Z}$ 上の元である)．各計算，すなわち加減乗除も $\mathbb{Z}/q\mathbb{Z}$ 上で行われる．シェア保有者 i のシェアは $f(i) \bmod q$ であり，ラグランジュ補間を用いて秘密 $s = f(0) \bmod q$ は

$$\begin{aligned} f(0) &= \sum_{i_l \in S} f(i_l)\lambda_{i_l,S}(0) \bmod q \\ &= \sum_{i_l \in S} f(i_l) \prod_{i_j \in S \setminus i_l} \frac{-i_j}{i_l - i_j} \bmod q \end{aligned}$$

として得ることができる．

$\mathbb{Z}/q\mathbb{Z}$ 上で構成された (k,n) 閾値法はいくつかの興味深い性質を有している．その一つは，$k-1$ 個のシェアを得たとしても，秘密 s の情報を全く得ることができないという性質である．実際，$k-1$ 個のシェアを集めたとしても，もう一つのシェアの値によっては，秘密 s は $\mathbb{Z}/q\mathbb{Z}$ のあらゆる値をとり得るということを示すことができる．実際，図 2 は $(3,n)$ 閾値法において，2 個のシェア $f(1)$ と $f(2)$ だけでは秘密を定めることができないことを直感的に示したものである．

一方，既に述べたように k 個のシェアを集めれば，秘密を得ることができる．この様な秘密分散方式は，完全であると呼ばれている．(k,n) 閾値法の場合，閾値 k やシェア保有者の総数 n をどのように取るかは，秘密を管理する組織などにより決定される．しかし，秘密 s の情報が一部漏れるという状態が存在しないことは，秘密 s を管理する側としては大変都合がよい．

なお，(k,n) 閾値法では，シェア $f(i) \bmod q$ は $\mathbb{Z}/q\mathbb{Z}$ の元である．例えば，q が 1000 ビットの場合，秘密 s もシェア $f(i) \bmod q$ も 1000 ビットとなり，秘密とシェアのサイズが等しい．実は，完全秘密分散を実現するためにはシェアのサイズは秘密 s のサイズ以上が必要となることが知られている．Shamir の (k,n) 閾値法は，完全秘密分散を達成しつつ，シェアのサイズが小さいという優れた方式である．

[2] $\mathbb{Z}/q\mathbb{Z}$ および $\mathbb{Z}/q\mathbb{Z}$ 上の演算については，[17, 20] などを参照のこと．

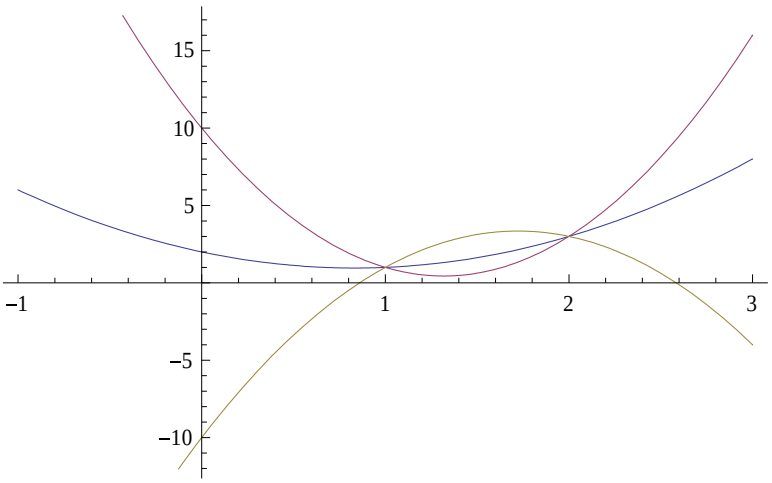

図 2: 閾値以下のシェアからわかること

3.2 行列を用いる手法

秘密分散を実現するために行列を用いる手法の考え方を説明する．行列の成分や演算は $\mathbb{Z}/q\mathbb{Z}$ 上で考えるものとする．秘密 s を U_1, U_2 に分散し，U_1, U_2 のシェア s_1, s_2 を全て集めたときにのみ復元できる方法を考える．まず,

$$G_1 = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

とする．秘密を $s \in \mathbb{Z}/q\mathbb{Z}$ とし，$r_1 \in \mathbb{Z}/q\mathbb{Z}$ をランダムに選ぶ．そして，ベクトル (s, r_1) を行列 G_1 の左からかけた

$$(s_1, s_2) = (s, r_1) \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

を，各々U_1, U_2 のシェアとする．すると，r_1 をランダムに選んだことにより，s_1 または s_2 だけでは s の情報を得ることができないことがわかる．また，s_1, s_2 の 2 つがそろえば s が求まることは，$s = s_2 - s_1 \bmod q$ よりわかる．同様に秘密 s を U_3, U_4, U_5 に分散し，全てのシェアを集めたときにのみ復元できる方法は,

$$G_2 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

とし，$r_2, r_3 \in \mathbb{Z}/q\mathbb{Z}$ をランダムに選んだ後，ベクトル (s, r_2, r_3) を行列 G_2 の左からかけた

$$(s_3, s_4, s_5) = (s, r_2, r_3) G_2$$

を各々U_3, U_4, U_5 のシェアとすればよい．さて，行列 G_1, G_2 を適切に組み合わせ

$$(s_1, s_2, s_3, s_4, s_5) = (s, r_1, r_2, r_3) \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

とすれば，$\{s_1, s_2, s_3, s_4, s_5\}$ の部分集合うち，$\{s_1, s_2\}$ を含むもの，もしくは $\{s_3, s_4, s_5\}$ を含むものから，秘密 s を復元できることがわかる．$\{s_1, s_2\}$ や $\{s_3, s_4, s_5\}$ は秘密 s を復元でき，かつ，1 個でもシェアが減ると秘密 s の情報を得ることができない[3]．前節の (k, n) 閾値法は秘密を復元するために必要なシェアの最小数は常に k であったので，本節の方法はより一般的な方法と位置づけることができる．

3.3 その他の秘密分散法

3.1 節や 3.2 節で紹介した考え方は，後述する属性ベース暗号等でも用いられる．さて，3.1 節で紹介した Shamir の (k, n) 閾値法は，完全秘密分散を実現していることは既に述べた．これに対し，$k-1$ 個以下のシェアから秘密 s の情報をある程度漏らしてもよいので，シェアのサイズを小さくすることは可能だろうか．これが可能なら，実用上のメリットは少なくない．このような方式の構成は可能で，ランプ型閾値法 [22] が広く知られている．ランプ型閾値法のパラメタは (k, L, n) の 3 つである．k は閾値でる．k 個のシェアからは秘密 s の情報を得ることができ，$k-L$ 個のシェアからは秘密 s の情報を全く得ることはできない．しかし，l 個 (ただし，$k-L+1 \leq l \leq k-1$) のシェアからは秘密 s の何らかの情報を得ることができる．正確には，l が大きくなるほど，秘密 s の情報を多く得ることができる．その一方，シェアのサイズを (k, n) 閾値法より小さくできるという利点がある．

ここまで，暗黙のうちに，ディーラは完全に信頼できるものとしてきた．しかし，ディーラがでたらめな値をシェアとして配布した場合は，当然復元は成功しない．これについての研究もなされており，検証可能秘密分散 [3] として知られている．

4 (k,n) 閾値暗号系

Blakley[2] や Shamir[12] の秘密分散法は，復号や署名生成に必要な秘密鍵の分割管理を志向していた．これらの方法では，一度復元されたら，秘密鍵は既知となる (秘密ではなくなる)．

本節では，暗号文の復号能力や署名生成能力に閾値法を適用する場合を考える．応用例をイメージするには，署名生成の例がわかりやすい．ある文書に対する署名を得るためには，一定の権限を有する者のうち少なくとも 2 名の同意 (協力) を得なくてはならないという場合を考える．ただし，その権限保有者が不在の場合なども考慮し，一定の権限を有する者は多めに (例えば 3 名) 設定し，そのうちの 2 名が協力すれば有効な署名を生成できるという方法があれば便利である．これは， Shamir の $(2, 3)$ 閾値法により秘密鍵 s のシェ

[3] 極小アクセス集合と呼ばれる．(k, n) 閾値法では，k 個のシェアから成る集合は全て極小アクセス集合になる．

アを 3 人に配布しておき，署名時に 3 人のうちの 2 人が協力して秘密鍵 s を復元することで実現できそうである．しかし，一度復元された s は秘密ではなくなるため，以後は,「3 人のうち 2 名の協力が必要である」という条件を実現できなくなる．そこで，3 人のうち 2 人が協力すれば，署名鍵を復元することなく署名の生成を可能とする方式が構成できればよい．

守秘に対しても同様のシナリオが考えられる．例えば，組織にとって重要な情報を暗号化した暗号文は，1 人では復号できないようにするというものである．この場合，暗号文の復号能力を 1 人ではなく，複数人に分割し，所定の数以上の協力が得られれば暗号文の復号ができ，そうでない場合は復号ができないということが求められる．

これを実現する (k, n) 閾値暗号系の研究は Desmedt らの研究 [4, 5] に端を発し，現在も活発に研究がなされている．本論文では，ElGamal 閾値復号法について説明する．

以後，話を簡潔にするために，ディーラは完全に信頼することとする．また，シェア保有者は所定の処理を正しく行う (プロトコルに従う) ものとする．ただし，シェア保有者は結託などにより，他人が有するシェアや秘密 s に関する情報を得ることを試みるものとする．

4.1 ElGamal 閾値復号法

復号権限を分散する例として，ElGamal 暗号の復号権限の分散の構成例を示す．まず，ElGamal 暗号について説明する．

鍵生成 $q|p-1$ となる素数 p, q，位数が q となる $g \in (\mathbb{Z}/p\mathbb{Z})^*$ を選ぶ[4]．更に，$s \in \mathbb{Z}/q\mathbb{Z}$ をランダムに選び，$h = g^s \bmod p$ とする．公開鍵を (p, q, g, h)，秘密鍵を s とする．

暗号化 平文 $m \in \langle g \rangle$ と公開鍵 (p, q, g, h) を入力とする．$r \in \mathbb{Z}/q\mathbb{Z}$ をランダムに選び，

$$(x, y) \quad = \quad (g^r \bmod p, h^r m \bmod p)$$

を暗号文とする．

復号 暗号文 (x, y) と秘密鍵 s を入力とし，

$$\frac{y}{x^s} \bmod p$$

を復号結果とする．

ElGamal 暗号の解説は，[17, 20] などを参照頂きたい．復号が成功する理由は，法 p で

$$\begin{aligned} \frac{y}{x^s} &\equiv \frac{h^r m}{(g^r)^s} \\ &\equiv \frac{(g^s)^r m}{g^{rs}} \\ &\equiv m \end{aligned}$$

[4] 位数，$\langle g \rangle$ については [17, 20] などを参照のこと．

となるからである．

ElGamal 暗号の場合，秘密の情報は秘密鍵 s である．これを (k,n) 閾値法を用いてシェアを生成し，n 人に渡すことは可能である．ElGamal 閾値復号法では，シェア保有者に部分的に復号をしてもらい，それらを集めて，最終的な復号結果を得ることを目指す．

鍵生成 $q|p-1$ となる素数 p,q，位数が q となる $g \in (\mathbb{Z}/p\mathbb{Z})^*$ を選ぶ．更に，$s \in \mathbb{Z}/q\mathbb{Z}$ をランダムに選び，$h = g^s \bmod p$ とする．公開鍵を (p,q,g,h)，秘密鍵を s とする．

シェアの生成 3.1 節で述べた Shamir の (k,n) 閾値法を用い，s を定数項とする秘密の $k-1$ 次多項式 $f(x)$ を生成し，シェア s_i を生成する．このシェアは，U_i に秘密裏に渡す．

部分復号 シェア s_i の保有者は，暗号文 (x,y) の第 1 成分 x に対し，

$$z_i \quad = \quad x^{s_i} \bmod p$$

を計算し，出力する．

復号 部分復号結果である z_i を集め，

$$z \quad = \quad \prod_{i \in S} z_i^{\prod_{j \in S \setminus i} \frac{-j}{i-j}} \bmod p$$

を計算する．そして，$y/z \bmod p$ を復号結果とする．

$y/z \bmod p$ により，平文を得ることができることは

$$\begin{aligned} \prod_{i \in S} z_i^{\prod_{j \in S \setminus i} \frac{-j}{i-j}} &= (g^r)^{\sum_{i \in S} s_i \prod_{j \in S \setminus i} \frac{-j}{i-j}} \\ &= g^{rs} \end{aligned}$$

により確認することができる．実は鍵生成時に，g として位数が q の元を選んでいる．位数が q であるということは，

$$g^q \equiv 1 \pmod{p} \tag{1}$$

を意味する．g の指数部の演算としては，a,b を整数とすると，加算 $(g^a \cdot g^b = g^{a+b})$ や乗算 $((g^a)^b = g^{ab})$ が考えられる．(1) 式より，g の指数部の演算は法 q 上の演算とみなすことができるので，3 節で述べた (k,n) 閾値法をそのまま適用できる．

5 属性ベース暗号への応用

最近活発に議論されているものの一つに，属性ベース暗号 [11, 9, 1] や関数型暗号 [10] がある．これらはいずれも公開鍵暗号系の一種とみなすことができるが，従来の公開鍵暗号などと異なる点が少なくない．

実は，秘密分散法が．これらの構築の際に使われている場合が多い．本節での解説のために必要な双線形性に関するいくつかの性質を述べた後，Sahai, Waters の方式 [11] を中心に紹介する．

5.1 双線形性

位数が素数 q の加法群 G と乗法群 G_T，及び写像 $e : G \times G \to G_T$ が与えられているとする．G や G_T 上の演算は効率よく計算可能であり，更に，写像 e については，

非縮退性 任意の $P, Q \in G$ に対し，$e(P, Q) = 1$ ならば $P = 0$ または $Q = 0$.

双線形性 任意の $P, Q \in G$ と任意の $a, b \in \mathbb{Z}$ に対し，

$$e(aP, bQ) \quad = \quad e(P, Q)^{ab}$$

となる．

計算可能性 写像 e は効率よく計算可能である．

の 3 つの性質を満たすとする．この時，e を双線形写像と呼ぶ．以下，加法群 G の生成元を P とする．なお，暗号用途に用いるには，

$(P, A(= aP), B(= bP), C(= cP))$ から，$e(P, P)^{abc}$ を計算することが難しい

とか，

$(P, A(= aP), B(= bP), C(= cP), Z)$ から，Z が $e(P, P)^{abc}$ か G_T からランダムに選ばれた元かを判定するのが難しい

という性質が成り立つものが選ばれる [16, 20].

5.2 Sahai, Waters の方式

属性ごとに識別子が $1, 2, \cdots$ と割り当てられているとする．例えば，ある組織では，複数のプロジェクト (これを属性とする) に社員が関与している．具体的には，社員 1，$\cdots$，社員 4 には表 1 のように関与しているプロジェクトが割り当てられているとする．

表 1: 社員と属性の対応

属性	プロジェクト	社員 1	社員 2	社員 3	社員 4
1	A	○	×	○	×
2	B	×	○	○	×
3	C	○	×	×	○
4	D	×	×	○	×
5	E	○	○	○	×
6	F	×	○	×	○

表 1 に示した例では，社員 1 はプロジェクト A, C, E に関与している．プロジェクト $A, B, \cdots$ は属性であり，各々識別子 $1, 2, \cdots$ に対応している．この時，プロジェクト A からプロジェクト D のうち，2 つ以上に関与している社員のみに情報を送りたい，逆にこの

条件を満たしていない社員にはその情報を知られたくないという場合を考える．送信者が，各社員がどのプロジェクトに関与しているか知っていれば，社員 1, 社員 3 の公開鍵を用いて暗号化し，その暗号文を (全員に) 送ってもよい．ただし，送り手 (暗号化を行う側) は社員のプロジェクトへの関与具合を知っている必要がある．

Sahai, Waters の方式 [11] は，閾値法をうまく利用してこれを実現する方法とみなすことができる．

パラメタ生成 e, G, G_T, q を生成する．次に，$t_1, \cdots, t_n \in \mathbb{Z}/q\mathbb{Z}$ をランダムに選び，$T_i = t_i P$ を生成する．また，$y \in \mathbb{Z}/q\mathbb{Z}$ をランダムに選び，$Y = e(P,P)^y$ とする．$(t_1, \cdots, t_n, y)$ がマスター秘密鍵であり，パラメタ $(e, G, G_T, q, P, \{T_i\}, Y)$ を公開する．

暗号化 パラメタ，平文 $m \in G_T$ 及び $W' \subset \{1, \cdots, n\}$ を入力とし，

1. 乱数 $s \in \mathbb{Z}/q\mathbb{Z}$ を選び，
2. W' および $(E', \{E_i\}_{i \in W'}) = (Y^s \cdot m, \{sT_i\}_{i \in W'})$ を暗号文

とする．

鍵生成 パラメタ，マスター秘密鍵，及び $W_j \subset \{1, \cdots, n\}$ を入力とする．まず，ランダムな $k-1$ 次多項式 $g_j(X)$ を選ぶ．ただし，$g_j(0) = y$ とする．次に

$$\{D_i = (\frac{g_j(i)}{t_i})P\}_{i \in W_j}$$

を生成し，$(W_j, \{D_i\}_{i \in W_j})$ を秘密鍵とする．これを社員 j に秘密裏に渡す．

復号 暗号文 $(W', E', \{E_i\}_{i \in W'})$ と秘密鍵 $(W_j, \{D_i\}_{i \in W_j})$ およびパラメタを入力とする．要素数が k となる集合 $S \subset W_j \cap W'$ が存在すればそれを選び，

$$\frac{E'}{\prod_{i \in S} e(D_i, E_i)^{\lambda_{i,S}(0)}} \tag{2}$$

を出力する．

復号が成功することは，(2) 式の分母が $e(P,P)^{ys}$ となることから確認できる．

表 1 の例に対応させれば，$n = 6, k = 2$ となる．暗号化時に指定する W' は $\{1,2,3,4\}$ である．社員 1 は $W_1 = \{1,3,5\}$ に対する秘密鍵を持っているので，$W_1 \cap W'$ の部分集合で，要素数が 2 となるように，$S = \{1,3\}$ とすれば復号できる．社員 2 は $W_2 \cap W' = \{2\}$ なので，復号することができない．

この方式では，送信者 (暗号化を行う者) は，社員のうちの誰が復号権限を有するか，今回の場合はプロジェクト A からプロジェクト D のうち，2 つ以上に関与している社員が誰であるかを知る必要がない．復号可能者に求められる条件である，「プロジェクト A からプロジェクト D のうち，2 つ以上に関与していることが復号できる条件」[5]であることをふまえ，暗号化しさえすればよい．

[5] ポリシーと呼ばれる．

[11] の安全性については，要素数が k となる集合 $S \subset W_j \cap W'$ が存在しない場合は，復号できないこと (平文 m の情報を得ることができないこと) を示す必要がある．注意すべきは，社員 2 と社員 4 が結託した場合，すなわち各々の秘密鍵を持ち寄った場合にも，$W' = \{1, 2, 3, 4\}$ として作られた暗号文から平文 m の情報を得ることができないことが求められる点である．直感的には社員 2 と社員 4 の秘密鍵生成に，独立な $k-1$ 次多項式 g_2, g_4 が用いられていることが寄与する．この方式の安全性証明等については [11] を参照いただきたい．

5.3 より細かな条件指定へ

更に詳細に条件を設定する場合を考える．例えば，表 1 において，プロジェクト A, C に属している，もしくはプロジェクト B, E, F に属している社員にのみ復号可能な方式の実現は可能だろうか．これは，復号可能となるための条件 (ポリシーと呼ばれる．以下，$(A \cap C) \cup (B \cap E \cap F)$ と書く) を暗号文に反映させる[6]ことで実現できる．これは，[1] や [13] など多くの構成方法が知られている．

一方，復号可能となるための条件を鍵生成時に，秘密鍵に反映させる方法[7]も [9] などの構成方法が知られている．

まず，ポリシー $(A \cap C) \cup (B \cap E \cap F)$ に対応する秘密 s のシェアとして，

1. A, C の両方のシェアが揃えば s を復元できる，または
2. B, E, F の全てのシェアを揃えれば s を復元できる

というものを考えよう．実は，3.2 節の方法を用いれば可能である．本節では，[9, 1] で用いられている，Shamir の (k, n) 閾値法を利用したシェアの構成方法を紹介する．方針としては，まず秘密分散法を用いて秘密 s を分割しシェアを生成する．その際，A, C の両方のシェア，もしくは B, E, F の全てのシェアを集めた場合に s を復元可能となるようにするというものである．(k, n) 閾値法では k, n というパラメタを自由に設定できる．まず，$X \cup Y$ を実現するためには，$(1, 2)$ 閾値法で秘密 s に対する 2 つのシェアを作り，X にシェア s_X を，Y にシェア s_Y を対応させればよい．一方，$X \cap Y$ を実現するためには，$(2, 2)$ 閾値法で秘密 s に対する 2 つのシェアを作り，X にシェア s_X を，Y にシェア s_Y を対応させればよい．あとはこれらの繰り返しである．$(A \cap C) \cup (B \cap E \cap F)$ に対応するシェアの作り方は，

1. $(1, 2)$ 閾値法で秘密 s のシェア s_X, s_Y を作る．実は，$s_X = s_Y = s$ である．
2. $(2, 2)$ 閾値法で秘密 s_X のシェア s_A, s_C を作り，A に s_A を，C に s_C を対応させる．
3. $(3, 3)$ 閾値法で秘密 s_Y のシェア s_B, s_E, s_F を作り，B, E, F に各々 s_B, s_E, s_F を対応させる．

[6] ciphertext policy と呼ばれる．
[7] key policy と呼ばれる．

とすればよい．実際，s_A, s_C を集めた場合，及び s_B, s_E, s_F を集めた場合に秘密 s を復元できることが確認できる．

この考え方を利用して，5.2 節の方式と組み合わせつつ (多少修正を加えることにより)，[9] が実現されている．

6 秘匿関数計算

秘密分散の応用，特に復号能力の制御に関する話題として，閾値復号法，属性ベース暗号を説明してきた．前節まで，秘密として，復号や署名時に用いる秘密の鍵を主に扱ってきたが，秘密として何らかの演算を施すべきデータを対象としたら何ができるだろうか．

(k, n) 閾値法において，秘密 s_A, s_B のシェアが，U_j に与えられていたとしよう．U_j が持つ秘密 s_i のシェアを $s_{i,j}$ と書くことにする．すると，$s_{A,j} + s_{B,j} \bmod q$ は $s_A + s_B \bmod q$ のシェアになっている．実際，シェアを作る際に利用する多項式を

$$\begin{aligned} f_A(x) &= a_{A,k-1}x^{k-1} + \cdots + a_{A,1}x + s_A \\ f_B(x) &= a_{B,k-1}x^{k-1} + \cdots + a_{B,1}x + s_B \end{aligned}$$

とすると，U_j のシェアは $f_A(j)$ と $f_B(j)$ であり，それらの和は

$$f_{A,B}(x) = (a_{A,k-1} + a_{B,k-1})x^{k-1} + \cdots + (a_{A,1} + a_{B,1})x + (s_A + s_B)$$

のシェア $f_{A,B}(j)$ に他ならない．よって，シェア $f_{A,B}(j)$ を k 個集めて復元できるのは $s_A + s_B$ となる．秘密 s_A, s_B を知ることなく，それらの和のシェアを得ることができる．

この手法を用いてシェアを k 個集めると，秘密の値である s_A, s_B を知ることなく，それらの和 $s_A + s_B$ のみを計算できることになる．ちなみに，シェアから積のみを計算する方法もあるが，和の場合ほど容易ではない．

複数の秘密を入力として，複数パーティにより，あらかじめ定められた演算の結果を得るという研究は，1980 年代より始まっている．例えば，[14, 15, 8] 等の研究に端を発し，様々な研究がなされている．

[8] は非常に汎用的な結果であり，AND と NOT の秘匿関数計算の実現している．AND と NOT を繰り返すことによりあらゆる演算は可能であるから，これを繰り返すことにより，入力を知られることなく，計算結果のみを得ることが可能となる．しかしながら，この方式は実用的とは言い難い．そこで，処理時間を減らすために，対象とする演算を限定する，特殊な秘密分散法を用いる，もしくは閾値 k などを限定するといったアプローチ [21, 19] がとられる場合も少なくない．

最近では，実用性を目的とし，3 者が協力することによる高速な秘匿関数計算に関する成果 [19] が発表されている．この研究では，高速な秘密分散方式を導入し，実用性を高めている．また，関係データベースにおける構造演算を，データベースサーバでデータを復元することなく行うといった成果 [18] も発表されている．ちなみに，[19, 18] のいずれも，情報処理学会の論文賞を受賞している．

参考文献

[1] J. Bethencourt, A. Sahai, B. Waters, Ciphertext‐policy attribute‐based encryption, IEEE Symposium on Security and Privacy 2007, pp.321-334, 2007.

[2] G. R. Blakley, Safeguarding cryptographic keys, Proc. of the National Computer Conference, Vol.48, pp.313-317, 1979.

[3] B. Chor, S. Goldwasser, S. Micali, B. Awerbuch, Verifiable secret sharing and achieving simultaneity in the presence of faults, Proc. of 26th Annual Symposium on Foundations of Computer Science (FOCS 1985), pp.383-395, 1985.

[4] Y. Desmedt, Society and Group Oriented Cryptography: a New Concept, Proc. of CRYPTO'87, LNCS 293, pp.120-127, 1987.

[5] Y. Desmedt, Y. Frankel, Threshold cryptosystems, Proc, of CRYPTO'89, LNCS 435, pp.307-315, 1989.

[6] Y. Desmedt, Y. Frankel, Perfect Homomorphic Zero-Knowledge Threshold Schemes over any Finite Abelian Group, SIAM J. Discrete Math. Vol.7, No.4, pp.667-679, 1994.

[7] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Trans. Information Theory, Vol.31, No.4, pp.469-472, 1985.

[8] O. Goldreich, S. Micali, A. Wigderson, How to Play Any Mental Game, or a Completeness Theorem for Protocols with an Honest Majority, Proc. of 19th annual ACM Symposium on Theory of Computing (STOC 1987), pp.218-229, 1987.

[9] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute‐Based Encryption for Fine‐Grained Access Control of Encrypted Data, Proc. of 13th ACM conference on Computer and Communications Security (CCS'06), pp.89-98, 2006.

[10] T. Okamoto, K. Takashima, Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption, Proc. of CRYPTO 2010, LNCS 6223, pp.191-208, 2010.

[11] A. Sahai, B. Waters, Fuzzy Identity‐Based Encryption, Proc. of EUROCRYPT2005, LNCS 3494, pp.457-473, 2005.

[12] A. Shamir, How to share a secret, Communications of the ACM, Vol.22, No.11, pp.612-613, 1979.

[13] B. Waters, Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization, Proc. of PKC 2011, LNCS 6571, pp.53-70, 2011.

[14] A. C. Yao, Protocols for Secure Computations, Proc. of 23rd Annual Symposium on Foundations of Computer Science (FOCS 1982), pp.160-164, 1982.

[15] A. C. Yao, How to generate and exchange secrets, Proc. of 27th Annual Symposium on Foundations of Computer Science (FOCS 1986), pp.162-167, 1986.

[16] 岡本栄司, 岡本健, 金山直樹, ペアリングに関する最近の研究動向, Fundamentals Review, Vol.1, No.1, pp.51-60, 電子情報通信学会基礎境界ソサイエティ, 2007.

[17] 佐々木良一 監修, 手塚悟 編著, 情報セキュリティの基礎, 共立出版, 2011.

[18] 志村正法, 宮崎邦彦, 西出隆志, 吉浦裕, 秘密分散データベースの構造演算を可能にするマルチパーティプロトコルを用いた関係代数演算, 情報処理学会論文誌, Vol.51, No.9, pp.1563-1578, 2010.

[19] 千田浩司, 五十嵐大, 濱田浩気, 高橋克巳, エラー検出可能な軽量 3 パーティ秘匿関数計算の提案と実装評価, 情報処理学会論文誌, Vol.52, No.9, pp.2674-2685, 2011.

[20] 辻井重男, 笠原正雄 編著, 暗号理論と楕円曲線, 森北出版株式会社, 2008.

[21] 保坂範和, 多田美奈子, 加藤岳久, 秘密分散法とその応用, 東芝レビュー, Vol.62, No.7, pp.23-26, 2007.

[22] 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.