Vol.2017-CSEC-77 No.15
Vol.2017-IOT-37 No.15
2017/5/26

パスワード付秘密分散法の秘匿検索への応用

辻下　健太郎[†1]　岩村　恵市[†1]

概要：クラウドなどの外部のサーバ内に保存するデータを秘匿化する際の方法として暗号化や秘密分散法といった手法がある.秘密分散法は故障耐性を持ち,秘匿化したまま演算を行うことができる準同型性を持つため,秘密分散法を用いたデータストレージや秘匿計算に関する研究が盛んに行われている.それに対して,秘密分散法を用いて直接保存したデータを検索するという秘匿検索を実現する技術は少ない.そこで,本論文では CSS2016 で中原らが提案したパスワード付き秘密分散法を用いた秘匿検索方式を提案する.さらに,検索者が効率よく検索を行うための機能の一つとして部分一致検索を付与した方式も提案する.

キーワード：秘匿検索　秘密分散法　部分一致検索

Application of password protected secret sharing scheme To searchable encryption

KENTARO TSUJISHITS[†1]　KEICHI IWAMURA[†1]

Abstract: There are methods such as encryption and secret sharing scheme, which stored conceal data in an external server. Since secret sharing scheme has failure tolerance and can compute conceal data, research on data storage and secret calculation using secret sharing scheme has been actively conducted. On the other hand, there are few techniques to realize secret search such as searching directly stored data using secret sharing scheme. In this paper, we propose a secret search using by password-protected secret sharing scheme proposed by Nakahara et al. in CSS2016. In addition, we also propose a method that can conduct partial match search in order to conduct efficient searches for searchers.

Keywords: secret search, secret sharing, partial match search

1. はじめに

近年,クラウドコンピューティングを利用したサービスの普及に伴い,ユーザが自前の装置でデータを保有するのではなく,オンライン上のサーバに保有することが数多く見受けられる.しかし,サーバに蓄積された情報の中には個人情報や企業内秘密情報などが含まれているため,データの秘匿化,秘匿化されたデータの検索,秘匿化されたデータの演算処理を行うことが求められている.その方法として秘密分散法,検索可能暗号,秘匿計算などが注目されている.

秘密分散法は１つの情報を複数の異なる情報に変換し,そのうちの一定数以上を集めることで元の情報を復元することができ,一定数未満であれば元の情報を復元することができないという手法である.これにより,災害等でデータの一部が使えなくなったとしてもそれが一定数以下であれば元の情報が復元でき,一定数以上の情報が漏えいしない限り情報漏洩は起こらないので安全なデータの運用を実現できる.また,秘密分散法には秘匿化したままデータの演算ができるという準同型性を持つため,安全に統計演算も行うことができる.そのため,秘密分散法を用いたデータストレージや,秘匿計算に関する研究はよく行われている.それに対して,秘密分散法を用いて直接保存したデータを検索するという秘匿検索を実現する技術は少ない.

検索可能暗号は情報を秘匿化した状態で検索を可能とする暗号であり共通鍵暗号方式と公開鍵暗号方式を利用する手法が知られており,現在よく研究されている.共通鍵暗号方式は秘密鍵をもつユーザのみが暗号化と検索を行うことができる.また,検索において高速性に優れ,計算量が少ないという特徴がある.しかし,データの所有者であるオーナとデータの検索者であるユーザは同一人物であるか,オーナが検索を許可したユーザに秘密鍵を配布する必要があり安全性が低下する.公開鍵暗号方式はデータの暗号化に公開鍵,検索に秘密鍵を用いて行うため,共通鍵暗号方式のようにオーナとユーザが同一人物である必要はない.また,[1]のように検索の委任のような様々な機能を付加することができる.しかし,公開鍵暗号を用いるため検索における計算量が多いという欠点がある.

本論文では秘密分散方式を秘匿検索に応用し,共通鍵暗号方式のように検索において高速性に優れ,公開鍵暗号方式よりも計算量が少なく,部分一致検索の機能を付与した秘匿検索手法を提案する.

本論文の構成を以下に示す.第 2 章で従来方式,第 3 章で秘匿検索を実現する提案方式及びその安全性などについて説明する.その後第 4 章で提案方式の拡張を行い,第 5 章で

†1 東京理科大学
Tokyo University of Science

Vol.2017-CSEC-77 No.15
Vol.2017-IOT-37 No.15
2017/5/26

まとめを行う.

2. 従来方式

2.1 秘密分散法

2.1.1 (k,n)閾値秘密分散法

次の二つの条件を満たす秘密分散法を,(k,n)閾値秘密分散法という.

(1) 任意の k 個の分散情報から,元の秘密情報 s を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 に関する情報は一切得ることはできない.

Shamir の提案した方法[2](以降,(k,n)Shamir 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) $\mathrm{s} < \mathrm{p}$かつ$\mathrm{n} < \mathrm{p}$である素数 p を選ぶ.

(2) $\mathrm{GF(p)}$の元から,異なる n 個の$x_i(\mathrm{i} = 0, \dots, \mathrm{n}-1)$を選び出し,ユーザ ID とする.

(3) $\mathrm{GF(p)}$の元から,$\mathrm{k}-1$個の乱数$a_j(j = 1,2, \dots, k-1)$を選んで,以下の式を生成する.

$$W_i = \mathrm{s} + a_1 x_i + a_2 x_i^2 + \cdots + a_{k-1} x_i^{k-1} \pmod{\mathrm{p}} \quad ①$$

(4) 上記式①のx_iに各ユーザ ID を代入して,分散情報W_iを計算し,各サーバに(x_i, W_i)を配布する.

[復元]

(1) 復元に用いるサーバを k 台選び,そのサーバの持つ(x_i, W_i)を受け取る.

(2) 分散式①に(x_i, W_i)を代入し,k 個の連立方程式を解くことで,秘密情報 s を得る.s の復元の際には,Lagrange の保管公式を用いると便利である.

2.1.2 (k,L,n)閾値ランプ型秘密分散法

(k,n)Shamir 法では,分散情報のサイズが$|s| \leqq |W_i|$となるため,容量効率がよくない.しかし,(k,L,n)閾値ランプ型秘密分散法(以降,ランプ型秘密分散法)は,ユーザが持つ分散情報のサイズを 1/L にすることができる[3].ランプ型秘密分散法は以下の三つの条件を満たす.

(1) 任意の k 個の分散情報から,元の秘密情報 s を復元することができる.

(2) 任意の$\mathrm{k}-\mathrm{t}$個$(1 \leq \mathrm{t} \leq \mathrm{L}-1)$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) $\mathrm{k}-\mathrm{L}$個以下の分散情報からは,秘密情報 s に関する情報は一切得ることはできない.

(k,n)Shamir 法を変更することでランプ型秘密分散法を実現できる.秘密情報を$\mathrm{s} = (s_0, s_1, \dots, s_{L-1})$とし,以下のように式を定めて,分散情報$W_i$を計算する.

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

復元の際は(k,n)Shamir 法と同様の手順で②の連立方程式を解き,$s_0, s_1, \dots, s_{L-1}$を得る.ランプ型秘密分散法は(k,n)Shamir 法と比較してデータ量を 1/L にすることができるが,$\mathrm{k}-\mathrm{L}$個以上の分散情報が流出すると,段階的に秘密情報に関するデータの漏えいが生じる.[4]

2.1.3 CSEC70 方式

CSEC70 方式は,秘密情報に乱数を乗じて秘匿化秘密情報を生成し,それを秘密分散する.秘匿乗算を行う際には,秘匿化秘密情報を一時的に復元してスカラー量として扱い,他の分散値と乗算を行う.これにより,乗算した際に多項式の次数は増加しないので,閾値を変化させずに秘匿乗算を行うことができる.以下,CSEC70 方式で用いる記号の定義,分散と復元,秘匿乗算,秘匿加減算について示す.以降においての四則演算は明示しない限り n 及び秘密情報 *s,a,b,c* より大きな素数 p を法としたものとする.例として演算$(a+b)(c+s)$は$(a+b)(c+s) \bmod \mathrm{p}$を意味する.また,CSEC70 方式での秘密情報は 0 を含まない$\mathrm{GF(p)}$上の値であり,用いられる乱数は 0 を除く$\mathrm{GF(p)}$上の一様乱数である.

[記号定義]

a, b, c:秘密情報

$\alpha_j, \beta_j, \gamma_j$:乱数

p:秘密情報$\mathrm{a, b, c} < \mathrm{p}$かつ$\mathrm{n} < \mathrm{p}$

$\overline{[x]_i}$:値 *x* に対するサーバS_iの保持する分散値

$[x]_i$:値 *x* に関連してサーバS_iが保持する分散値集合

[分散]

(1) ディーラは k 個の乱数$\alpha_j(j = 0, \dots, k-1)$を生成し,$\alpha = \prod_{j=0}^{k-1} \alpha_j$を計算する.

(2) ディーラはαaを計算し,$\alpha a, \alpha_0, \dots, \alpha_{k-1}$を(k,n)Shamir 法で n 台のサーバに分散する.

(3) サーバ$S_i(i = 0, \dots, \mathrm{n}-1)$は秘密情報 *a* に対する分散値集合として$[a]_i \coloneqq (\overline{[\alpha a]_i}, \overline{[\alpha_0]_i}, \dots, \overline{[\alpha_{k-1}]_i})$を保持する.

[復元]

(1) ユーザは k 台のサーバS_jから$[a]_j$を受け取る.

(2) ユーザはサーバS_jから受け取った$[a]_j \coloneqq (\overline{[\alpha a]_j}, \overline{[\alpha_0]_j}, \dots, \overline{[\alpha_{k-1}]_j})$を用いて$\alpha a, \alpha_0, \dots, \alpha_{k-1}$を復元し,以下より *a* を復元する.

$$\alpha = \prod_{j=0}^{k-1} \alpha_j$$

$$\alpha \mathrm{a} \times \alpha^{-1} = a$$

[秘匿乗除算]

$c = ab$を計算するとき,各サーバS_iは[分散]で計算された秘密情報 *a,b* に関する分散値集合$[a]_i$と$[b]_i$を持ち,以下の手順により分散値集合$[c]_i$を計算する.また,(1)におけるS_0は任意に定められる.ただし,$a \neq 0$とする.

(1) サーバS_0は任意の k 台のサーバ$S_j(\mathrm{j} = 0, \dots, \mathrm{k}-1)$から$\overline{[\alpha a]_j}$を受け取る.

Vol.2017-CSEC-77 No.15
Vol.2017-IOT-37 No.15
2017/5/26

(2) サーバS_0はαaを復元し,すべてのサーバへαaを送信する.

(3) 全てのサーバ$S_i(i=0,\dots,n-1)$はそれぞれ$\overline{[\alpha\beta ab]_i} = \alpha a \times \overline{[\beta b]_i}$を計算する.

(4) k 台のサーバ$S_j(\mathrm{j}=0,\dots,\mathrm{k}-1)$は任意の k 台のサーバから各々$\left(\overline{[\alpha_0]_j},\dots,\overline{[\alpha_{k-1}]_j}\right),\left(\overline{[\beta_0]_j},\dots,\overline{[\beta_{k-1}]_j}\right)$を集め,$\alpha_j,\beta_j$を復元する.

(5) k 台のサーバS_jは$\alpha_j\beta_j$を計算して,n 台のサーバに分散する.

(6) 全サーバS_iはそれぞれ秘密情報 c の分散値集合として$[c]_i := (\overline{[\alpha\beta ab]_i},\overline{[\alpha_0\beta_0]_i},\dots,\overline{[\alpha_{k-1}\beta_{k-1}]_i})$を保持する.

秘匿除算を行う場合は,(3)における計算を$\overline{[\beta b/\alpha a]_i} = \overline{[\beta b]_i}/\alpha a$に,(5)における$\alpha_j\beta_j$を$\beta_j/\alpha_j$とすることによって実現できる.

[秘匿加減算]

$c=a\pm b$を計算するときに,各サーバS_iは 2.2.2 で計算された秘密情報a,bに関する分散値集合$[a]_i$と$[b]_i$を持ち,以下の手順により分散値集合$[c]_i$を計算する.

(1) k 台のサーバ$S_j(\mathrm{j}=0,\dots,\mathrm{k}-1)$は任意の k 台のサーバからそれぞれ$\left(\overline{[\alpha_0]_j},\dots,\overline{[\alpha_{k-1}]_j}\right),\left(\overline{[\beta_0]_j},\dots,\overline{[\beta_{k-1}]_j}\right)$を集め,$\alpha_j,\beta_j$を復元し,乱数$\gamma_j$を生成して,$S_0$に$\gamma_j/\alpha_j,\gamma_j/\beta_j$を送信する.

(2) サーバS_0は以下よりγ/α,γ/βを復元し,全てのサーバに送信する.

$$\frac{\gamma}{\alpha}=\prod_{j=0}^{k-1}\frac{\gamma_j}{\alpha_j}$$

$$\frac{\gamma}{\beta}=\prod_{j=0}^{k-1}\frac{\gamma_j}{\beta_j}$$

(3) 全てのサーバ$S_i(i=0,\dots,n-1)$はそれぞれ以下の計算を行う.

$$\overline{[\gamma(a\pm b)]_i}=\frac{\gamma}{\alpha}\overline{[\alpha a]_i}\pm\frac{\gamma}{\beta}\overline{[\beta b]_i}$$

(4) k 台のサーバ$S_j(\mathrm{j}=0,\dots,\mathrm{k}-1)$は$\gamma_j$を n 台のサーバに分散する.

(5) 全てのサーバ$S_i(i=0,\dots,n-1)$は分散値集合$[c]_i = (\overline{[\gamma(a\pm b)]_i},\overline{[\gamma_0]_i},\dots,\overline{[\gamma_{k-1}]_i})$を保持する.

3. 提案方式

3.1 提案方式の概要

本章では,[10]で提案されたパスワード付き秘密分散法を検索可能暗号に拡張させる.具体的には,パスワード付き秘密分散法で用いられるパスワードを検索キーワードとみなし,ユーザ認証の機能ではなくキーワードの正誤判定として利用する.以降[11]を参考に検索可能秘密分散法としてシステムモデル,アルゴリズム,安全性について示す.

3.2 システムモデル

検索可能秘密分散法では,ドキュメント群$\mathrm{D}=\{D_1,\dots,D_M\}$を持つオーナ,キーワード K'を持つドキュメントを検索したいユーザ,n 台のサーバ$S_i(i=0,\dots,n-1)$が存在する.共通鍵暗号方式を用いた検索可能暗号との違いは,オーナとユーザは同一人物でなくてもよいという点である.また,各ドキュメントD_mは識別子Did_mを持つまた,各ドキュメントD_mは識別子Did_mを持ち,この識別子の集合を$D_{ID}=\{Did_1,\dots,Did_m\}$とし,これをドキュメント辞書と呼ぶ. ドキュメントの識別子Did_mはシリアル番号などでもよく, ドキュメント辞書は公開情報とする.また,検索可能なキーワードの集合を$\Delta=\{K_1,\dots,K_L\}$とし,Δを辞書と呼ぶ.さらに,キーワード$K_l\in\Delta$によるドキュメントの検索結果を$Did(K_l)=\{Did_m|K_l\in D_m\}$とする.

各ドキュメントD_mは共通鍵暗号方式で暗号化し,暗号化ドキュメント群$\mathrm{C}=\{C_1,\dots,C_M\}$として 1 台のサーバ$S_0$に保存する(保存する場所が分かっていれば複数のサーバに分散保存してもよい).ただし,暗号化に用いた鍵$\mathrm{Y}=\{Y_1,\dots,Y_M\}$とドキュメント識別子$Did_m$は秘密分散して後述のように全サーバに保存する.

3.3 アルゴリズム

検索可能秘密分散法のアルゴリズムは Buildindex(D, Δ),Trapdoor(K'),Search(I, $T_{K'}$)の 3 つであり,それぞれインデックス生成,トラップドア生成,検索の役割がある.

3.3.1 インデックス生成: Buildindex(D, Δ)

(1) オーナは各ドキュメントD_mを鍵Y_mを用いて共通鍵暗号で暗号化し,暗号化ドキュメントC_mを生成する.その後,暗号化ドキュメント群$\mathrm{C}=\{C_1,\dots,C_M\}$を 1 台のサーバ$S_0$に送信する.

(2) オーナは各ドキュメントD_mに対して以下の(3),(4)を行う.なお,ドキュメントD_mが持つキーワード群を$K(D_m)=\{K_l|K_l\in D_m\}$とする.

(3) オーナはドキュメントD_m中の各キーワードK_lに対して,それぞれ以下の手順で分散を行う.

① オーナは k 個の乱数$\alpha_{l,j}(\mathrm{j}=0,\dots,\mathrm{k}-1)$を生成し,$\alpha_l=\prod_{j=0}^{k-1}\alpha_{l,j}$を計算する.同様に,k 個の乱数$\delta_{l,j}(\mathrm{j}=0,\dots,\mathrm{k}-1)$を生成し,$\delta_l=\prod_{j=0}^{k-1}\alpha_{l,j}$を計算する.

② オーナはキーワードK_lに対して$\alpha_l K_l$を計算し,$\alpha_l K_l,\alpha_{l,0},\dots,\alpha_{l,k-1}$をそれぞれ(k,n)Shamir 法で分散し,キーワードK_lに対する分散値集合として$[K_l]_i^{(1)} := (\overline{[\alpha_l K_l]_i},\overline{[\alpha_{l,0}]_i},\dots,\overline{[\alpha_{l,k-1}]_i})$を生成する.同様に,$\delta_l,\delta_{l,0},\dots,\delta_{l,k-1}$をそれぞれ(k,n)Shamir 法で分散し,1 に対する分散値集合として$[1_l]_i^{(1)} := (\overline{[\delta_l]_i},\overline{[\delta_{l,0}]_i},\dots,\overline{[\delta_{l,k-1}]_i})$を生成する.

(4) オーナは前記$[K_l]_i^{(1)}$と$[1_l]_i^{(1)}$を$K(D_m)$に対応させて,$[K(D_m)]_i := ([K_l]_i^{(1)},[1]_i^{(1)}|K_l\in D_m)$を生成する.

(5) オーナは各ドキュメント識別子Did_mと鍵Y_mをそれぞ

Vol.2017-CSEC-77 No.15
Vol.2017-IOT-37 No.15
2017/5/26

れ(k,n)Shamir 法で分散し,$\overline{[Did_m]_i}, \overline{[Y_m]_i}$を生成する.

(6) オーナは n 台のサーバ$S_i(i = 0, \dots, n-1)$にインデックス

$I = \{(\overline{[Did_1]_i}, [K(D_1)]_i, \overline{[Y_1]_i}), \dots, (\overline{[Did_M]_i}, [K(D_M)]_i, \overline{[Y_M]_i})\}$を送信する.ただし,インデックス中の$(\overline{[Did_m]_i}, [K(D_m)]_i, \overline{[Y_m]_i})$を識別子集合と呼ぶ.

3.3.2 トラップドア生成: Trapdoor(K')

(1) 検索者であるユーザは k 個の乱数$\beta_j(j = 0, \dots, k-1)$を生成し,$\beta = \prod_{j=0}^{k-1} \beta_j$を計算する.同様に,k 個の乱数$\eta_j(j = 0, \dots, k-1)$を生成し,$\eta = \prod_{j=0}^{k-1} \eta_j$を計算する.

(2) ユーザは検索キーワード K'に対して$\beta K'$を計算し,$\beta K', \beta_0, \dots, \beta_{k-1}$をそれぞれ(k,n)Shamir 法で分散し,検索キーワード K'に対する分散値集合として$[K']_i^{(1)} := (\overline{[\beta K']_i}, \overline{[\beta_0]_i}, \dots, \overline{[\beta_{k-1}]_i})$を生成する.同様に,$\eta, \eta_0, \dots, \eta_{k-1}$をそれぞれ(k,n)Shamir 法で分散し,1 に対する分散値集合として$[1]_i^{(2)} := (\overline{[\eta]_i}, \overline{[\eta_0]_i}, \dots, \overline{[\eta_{k-1}]_i})$を生成する.

(3) ユーザは k 台のサーバ$S_j(j = 0, \dots, k-1)$にトラップドア$T_{K'} = ([K']_i^{(1)}, [1]_i^{(2)})$を送信する.

3.3.3 検索: Search(I, $T_{K'}$)

(1) k 台のサーバ$S_j(j = 0, \dots, k-1)$は送られてきた$T_{K'}$と保存されているインデックス I を利用して以下の(2),(3)を全インデックスに分繰り返す.

(2) k 台のサーバは全$[K(D_m)]_j$に対して以下の操作を行う.

① k 台のサーバは$[K_l]_j^{(1)}$と$[1_l]_j^{(2)}$について秘匿乗算を行い,$[K_l]_j^{(2)} := (\overline{[\alpha_l \eta K_l]_j}, \overline{[\alpha_{l,0}\eta_0]_j}, \dots, \overline{[\alpha_{l,k-1}\eta_0]_j})$を生成する.

② k 台のサーバは同様に$[K']_j^{(1)}$と$[1_l]_j^{(1)}$について秘匿乗算を行い,$[K']_j^{(2)} := (\overline{[\beta\delta K_l]_j}, \overline{[\beta_0\delta_0]_j}, \dots, \overline{[\beta_0\delta_{k-1}]_j})$を生成する.

③ k 台のサーバは$[K_l]_j^{(2)}$と$[K']_j^{(2)}$について秘匿減算を行い,$\overline{[\gamma_l(K_l - K')]_j}$を生成する.ただし,秘匿検索において$\overline{[\gamma_{l,0}]_j}, \dots, \overline{[\gamma_{l,k-1}]_j}$は復元に用いられることはないので$\overline{[\gamma_{l,0}]_j}, \dots, \overline{[\gamma_{l,k-1}]_j}$に関する処理は省略される.

(3) k 台のサーバは$\overline{[\gamma_l(K_l - K')]_j}$を復元し,復元結果が 0 になった場合,それに対応する$(\overline{[Did_m]_i}, \overline{[Y_m(K_l)]_j})$をユーザに送信する.ただし,$\overline{[Y_m(K_l)]_j}$は以下のように生成する.

$$\overline{[Y_m(K_l)]_j} := \overline{[\gamma_l(K_l - K')]_j} + \overline{[Y_m]_j} + \sum_{i=0}^{k-1} f_{0,i}(j)$$

(4) ユーザは$\overline{[Did_m]_i}$を復元する.その後,復元したドキュメント識別子に加え,ダミーの識別子をドキュメント辞書$D_{ID} = \{Did_1, \dots, Did_m\}$の中から任意に選び,サーバ$S_0$に送信する.ただし,この時選択するダミーのドキュメント識別子は復元したドキュメント識別子以外から選ぶ.

(5) サーバS_0は送られてきたドキュメント識別子に対応する(Did_m, C_m)をユーザに送信する.

(6) ユーザは送られてきた$\overline{[Y_m]_j}$を復元してY_mを得,それに対応するC_mを復号してドキュメント$D_m(m \in Did(K_l))$を得る.

3.3.4 シャッフル

検索可能秘密分散法では,キーワードの一致判定をシステム側で行っている.これより,キーワードが一致して$\gamma_l(K_l - K') = 0$となったとき,システム側はそれに対応するインデックス中の識別子集合を知ることができ,システムは各識別子集合に対する頻度分析を行うことができる.この対策として,以下のように識別子集合単位のシャッフルを行う.ただし,簡単のため$\overline{[Did_m]_i}$と$\overline{[Y_m]_i}(m = 1, \dots, M)$をまとめてシャッフルを説明する.また,更新器となるサーバの順序は予め決まっているとする.

(1) 全てのサーバ$S_i(i = 0, \dots, n-1)$は,乱数r_1, r_2を生成し,各々$\overline{[Did_m]_i}$と$\overline{[Y_m]_i}$に加算し,$\overline{[Did_m + r_1]_i}^{(0)}$と$\overline{[Y_m + r_2]_i}^{(0)}$を生成する.

(2) 全てのサーバ$S_i(i = 0, \dots, n-1)$は$\overline{[Did_m + r_1]_i}^{(0)}$と$\overline{[Y_m + r_2]_i}^{(0)}$をまとめて$\{\overline{[Did_m + r_1]_i}^{(0)}, \overline{[Y_m + r_2]_i}^{(0)}\}$として,送信元のサーバを識別できるタグをつけて,1 つのサーバ（以降,更新器）に送信する.

(3) 更新器は$\overline{[Did_m + r_1]_i}^{(0)}$と$\overline{[Y_m + r_2]_i}^{(0)}$の m 毎に,以下のように分散値を更新する.ただし,$\overline{[Did_m + r_1]_i}^{(0)}$と$\overline{[Y_m + r_2]_i}^{(0)}$に加える$\overline{[0]_i}$は異なる.

$$\overline{[Did_m + r_1]_i}^{(t+1)} = \overline{[Did_m + r_1]_i}^{(t)} + \overline{[0]_i}$$

$$\overline{[Y_m + r_2]_i}^{(t+1)} = \overline{[Y_m + r_2]_i}^{(t)} + \overline{[0]_i}$$

(4) 更新器は,$\{\overline{[Did_m + r_1]_i}^{(t+1)}, \overline{[Y_m + r_2]_i}^{(t+1)}\}(m = 1, \dots, M)$をシャッフルして,次のサーバに全$\{\overline{[Did_{m'} + r_1]_i}^{(t+1)}, \overline{[Y_{m'} + r_2]_i}^{(t+1)}\}(m' = 1', \dots, M')$を送信する（$m$と$m'$の対応はランダム）.

(5) 次のサーバは更新器として(3)(4)を実行する.

(6) 最後のサーバは(3)(4)を実行した後,タグを用いて送信元のサーバに$\{\overline{[Did_{m'}+r_1]_i}^{(n)}, \overline{[Y_{m'}+r_2]_i}^{(n)}\}$を戻す.

(7) 元のサーバは(1)において加えた乱数r_1, r_2を$\overline{[Did_{m'}+r_1]_i}^{(n)}, \overline{[Y_{m'}+r_2]_i}^{(n)}$から引くことによって,新たな分散値$\overline{[Did_{m''}]_i}$と$\overline{[Y_{m''}]_i}(m''=1'',\dots,M'')$を得る.

識別子集合は$\overline{[Did_{m''}]_i}$と$\overline{[Y_{m''}]_i}$以外に,複数の$[K(D_m)]_i$を含み,$[K(D_m)]_i$は分散値集合である$[K_l]_i{}^{(1)}$と$[1_l]_i^{(1)}$からなる.よって,$[K_l]_i{}^{(1)}$,$[1_l]_i^{(1)}$中の各分散値に対してもまとめて上記シャッフルを行えば,識別子集合単位でのシャッフルが実現できる.

3.4 安全性

まず 3.2.1~3.2.3 で提案した検索可能秘密分散法の安全性について議論する.ただし,システムは独立に管理されているサーバからなり,全サーバはプロトコルに従って正しい動作を行うとする.ただし,k-1 台までのサーバは攻撃者にその情報を知られる可能性があるとする.また,ここでは簡単のため 1 つの識別子集合中の$[K(D_m)]_i$に含まれるキーワードの数は一つとする（一つ以上のキーワードがある場合,1 つのドキュメント識別子に対して複数の識別子集合を設定する）.これによって,識別子集合のサイズなどによって区別できないとする.以上の前提の下,以下の攻撃を考える.

攻撃者 1 :システムが攻撃者となる場合である.前提により,k-1 台のサーバ情報を得ることができ,その情報を用いて各ドキュメントが持つキーワード及びドキュメント自体を得ようとする.さらに,検索者が検索しているキーワード及びその検索傾向を知ろうとする.

攻撃者 2 :検索者が自分の選択したキーワードをシステムに入力し,そのキーワードに対応するドキュメントがあるかを検索する場合を考える.この検索者が攻撃者となる場合を攻撃者 2 とし,登録されているキーワード,及びそのキーワードに対応するドキュメントを得ようとする.

攻撃者 3 :オーナが攻撃者となる場合であって,攻撃者 1 が知る情報に加えて 3.2.1 におけるすべての情報を知る.検索者が検索しているキーワード及びその検索傾向を知ろうとする.

3.4.1 攻撃者 1 に対する安全性

検索可能秘密分散法では,インデックスとして保存される全ての情報が秘密分散されているため,k-1 台のサーバ情報を得ることができても,ドキュメント識別子,そのキーワード,復号鍵に関する情報は全く得られない.ただし,識別子集合毎に区別されて保存されている場合,登録されている識別子集合の数が,公開されているドキュメント辞書中のドキュメントの数を超えている場合,1 つのドキュメントが 2 つ以上のキーワードをもつ場合があることを知るが,どのドキュメントが 2 つ以上のキーワードを持つかはわからない.また,同じキーワードが異なるドキュメントで用いられていても,異なる乱数を用いて秘匿化秘密情報を生成するので同じ分散値になることはなく,識別子集合中の同じキーワードは判定できない.さらに,演算過程で秘匿化秘密情報が一時的に復元されたとしても乱数が異なるため,キーワードが同じであるかどうか攻撃者は判定できず,部分的な情報も得られない.よって,保存されているインデックスから攻撃者 1 が得られる情報は 2 つ以上のキーワードをもつドキュメントがあるかないかだけである.

次に,検索者がトラップドアを生成し,システムに入力した場合を考える.この場合,トラップドアに含まれる全ての情報も秘密分散されており,トラップドアは秘密分散法を用いて確率的に生成されるので,検索者の入力について攻撃者 1 は何の情報も得ない.

次に,検索処理を行ったとき,k 台のサーバは$\overline{[\gamma_l(K_l-K')]_j}$の復元結果が 0 となる識別子集合を知る.しかし,そのドキュメント識別子はユーザが復元するため,システムはその識別子集合以外の情報を得ない.特に,検索が行われるたびにシャッフルを行うならば,ある検索者が同じキーワードを用いて検索が繰り返したとしても,復元結果が 0 となる識別子集合は更新されているため毎回異なり,識別子集合によって検索傾向を把握することができない.

ただし,C_mを保存するサーバS_0はユーザから指定されたDid_mを知る.しかし,ユーザから指定されるDid_mにはダミーが入っているのでどのドキュメントがユーザの検索したいものかを一意に特定することはできず,頻度分析を困難にできる.例えば,ユーザが必ず指定するダミーのドキュメント識別子があれば,ユーザが検索するドキュメント識別子やその検索頻度などをシステムに誤認識させることもできる.また,ダミーの数を増すことによってその安全を増大させていくことができる.

以上より,システムは登録されている情報及び検索者から送られてきた情報から有用な情報を得ることができず,かつ検索結果から検索の傾向などを得ることも困難と言える.

3.4.2 攻撃者 2 に対する安全性

一般に,共通鍵暗号を用いた検索可能暗号では秘密情報の登録者と検索者は同一人物であるので,検索者に対する安全性は考慮されず,システムのみを攻撃者（攻撃者 1）としてその安全性を評価した.検索可能秘密分散法では登録者と検索者を同じとしないため,検索者が攻撃者となる場合を考える必要がある.

特に, 検索者は 3.2.3(4)でダミーのドキュメント識別子を送ることによって，それに対応する暗号化ドキュメントを得るので，要求した以上の情報を得る可能性がある．しかし，ダミーのドキュメントは検索者が指定したキーワードを含まないので，3.2.3(3)において計算されるパスワード付秘密分散を正しく復元できない．すなわち，送られた暗号化ドキュメントを復号する鍵を正しく復元できない．よって，検索者が攻撃者となっても要求した以上の情報を得ることはできない．

また，検索者と登録者が異なる場合，検索者がキーワードを全数探索することによって，一致したキーワードに対応するドキュメントを得ることができるという問題がある．キーワードはドキュメントに対して不変なので，パスワードのように更新することは困難である．よって，以下のような対策を考える．

対策１：パスワードを設定し，そのパスワードとキーワードを知る者だけが検索成功とする．

例えば，識別子集合に登録者が定めたパスワードの分散値も含め，3.2.3(3)においてパスワードとキーワードの両方が一致しない限り，検索成功としない．この場合，前述したようにパスワードは定期的に更新できるので，長期間の安全性を保証できる．また，これによって検索者を登録者が指定できる．特に，登録者がパスワードを自分だけで使う場合，共通鍵暗号を用いた検索可能暗号と同様に検索者＝登録者とできる．

対策２：キーワードを全数探索に耐える長さとし，検索者の検索回数を有限回に抑える．

対策１では検索者が限定されてしまうので，誰でも検索可能という検索可能秘密分散法の良さが生かせない．よって，理論的ではなく実用的な安全性を確保するために，キーワードのサイズを 100bit 以上の全数探索に耐えるとされる長さとして，検索者からの検索回数を数万回程度に抑える．

対策３：キーワード辞書を公開とする．

検索可能秘密分散を Wikipedia のように誰でも登録でき，誰でも検索できる仕組みとしたい場合，キーワード自体が秘密ではないため，公開することによってキーワードに対する攻撃自体を無効とする．

公開の Wikipedia の仕組みと比較した場合，公開の仕組みでは登録者が登録する情報をシステムも含め誰でも知ることができる．また，ドキュメントを保存しているサーバが攻撃されると情報が漏洩する．また，システムは誰がどのドキュメントを検索しているか容易に知ることができる．それに対して，検索可能秘密分散法は k 台のサーバの情報が漏洩しない限り安全であり，攻撃者１で説明したようにシステムに検索情報は漏洩しない．よって，キーワードは秘密でなく誰でも使えるが，誰が何を検索しているかを秘密にしたいという応用に有効である．

3.4.3 攻撃者 3 に対する安全性

攻撃者 3 は 3.2.1 におけるインデックス情報を全て知る．しかし,攻撃者 3 は検索に関与しないため，攻撃者 1 と同様に k-1 台のサーバ情報しか得ることができない．また，シャッフルが行われると，登録時の識別子集合とシャッフル後の識別子集合は 3.3.4 に示すように全く対応が取れなくなるため，識別子集合から登録情報を知ることができなくなる．

よって，攻撃者３は攻撃者１と同様に，検索者から送られてきた情報から有用な情報を得ることができず，かつ検索結果から検索の傾向などを得ることも困難と言える．

3.5 共通鍵暗号方式との比較

共通鍵暗号を用いる検索可能暗号と検索可能秘密分散法との大きな違いは検索者＝登録者に限定されず，様々な使い方ができるという点であるが，それ以外の違いを詳細に検討する．

3.5.1 インデックスの大きさについて

共通鍵暗号方式と検索可能秘密分散法のインデックスの大きさを比較するため,以下のように定義する．

M:インデックス内のドキュメント ID の総数
N:インデックス内のキーワードの総数
A:ドキュメント ID のサイズ
B:キーワードのサイズ
R:提案方式の秘匿検索で用いる乱数のサイズ
Y:ドキュメントの暗号化に用いる乱数のサイズ

ただし,提案方式のドキュメント ID とキーワードは秘密分散されてサーバに保存されているが,ここでは秘密情報のサイズと分散情報のサイズは同じとする．

共通鍵暗号方式では,オーナは自身が持つ秘密鍵とキーワードを用いてインデックス生成を行うため,インデックスの大きさは$MA + NB$となる．

提案方式では,一つのキーワードに対してキーワードの分散情報のほかに,キーワードにかかっている乱数に関する k 種類の分散情報と秘匿演算に必要な 1 の分散情報があるため,共通鍵暗号方式に比べて,キーワード一つ当たり$(2k + 1)R$大きくなる.また,提案方式では,複数のユーザを前提としているため,ドキュメントの暗号化に用いた秘密鍵もインデックス内に存在するので MY だけ大きくなる.以上より,検索可能秘密分散法のインデックスの大きさは$M(A + Y) + N(B + (2k + 1)R)$となる.よって,検索可能秘密分散法のインデックスの大きさは共通鍵暗号方式に比べて$MY + N(2k + 1)R$だけ大きくなる．

Vol.2017-CSEC-77 No.15
Vol.2017-IOT-37 No.15
2017/5/26

3.5.2 許容可能な情報漏洩について

共通鍵暗号を用いた秘匿検索では安全性を検討する際にドキュメントの追加およびキーワードの検索を行った際に必ず流出する情報を定義し[11],それ以上の情報が流出しない方式を安全としている場合が多い.本論文では,共通鍵暗号を用いた既存方式と提案方式の流出情報を比較するために, 既存方式で漏洩する以下の情報について検討する.

① ドキュメントD_mの長さ$|D_m|$
② あるトラップドア T(K')でキーワードの一致判定を行った際に出力される検索結果$Did(K')$
③ 検索者がi回目の検索に使用したキーワードK'とj回目に検索に使用したキーワードK"が等しいか否か
④ ドキュメントD_mと対応するキーワードの数
⑤ Forward Privacyを満たすかどうか

表1に,共通鍵暗号を用いた既存方式[14][15][16][17][18][19]と提案方式が②～⑤の情報について流出するかどうかを表したものを示す.ただし,表中の○は流出しないことを示し,×は流出することを表す.また,①は必ず流出するため省略する

表1 既存方式と提案方式の流出情報の対応

	②	③	④	⑤
[14]	×	×	○	×
[15]	○	○	×	×
[16]	×	○	×	×
[17]	×	○	×	×
[18]	×	×	○	×
[19]	×	△	○	○
提案方式	△	○	○	○

表1より,②～⑤のどれか一つについて対策している方式は多いが,複数の情報漏洩について対策している方式は既存方式の[15],[19],提案方式の三つである.したがって,提案方式は他の既存方式に比べ安全な方式といえる.

4. 部分一致検索

検索可能暗号では,キーワードの一致判定を行うとき,トラップドアに指定するキーワードと暗号化タグ内のキーワードが完全に一致したときに検索成功とする完全一致検索である場合が多い.これに対し,暗号化タグ内のキーワードの一部が一致していたら検索成功とする方式も提案されており[12][13],このような検索を部分一致検索と呼ぶ.以下,3.3 に示した検索秘密分散法を用いた秘匿検索手法の一部を変更して部分一致検索を実現したものを示す.

[インデックス生成:Buildindex(D, Δ)]

(1) オーナは各ドキュメントD_mを鍵Y_mを用いて共通鍵暗号で暗号化し,暗号化ドキュメントC_mを生成する.その後,暗号化ドキュメント群$C = \{C_1, \dots, C_M\}$を 1 台のサーバS_0に送信する.

(2) オーナは各ドキュメントD_mに対して以下の 3,4 を行う.なお,ドキュメントD_mが持つキーワード群を$K(D_m) = \{K_l | K_l \in D_m\}$とする.

(3) オーナはドキュメントD_m中の各キーワードK_lに対して,それぞれ以下の手順で分散を行う.

① ユーザはk個の乱数$\alpha_{l,j}(j = 0, \dots, k-1)$を生成し,$\alpha_l = \prod_{j=0}^{k-1} \alpha_{l,j}$を計算する.同様に,k 個の乱数$\delta_{l,j}(j = 0, \dots, k-1)$を生成し,$\delta_l = \prod_{j=0}^{k-1} \delta_{l,j}$を計算する.

② キーワードK_lの文字数を L とすると,K_lを L 分割し,$K_l = K_{l,1} \| K_{l,2} \| \cdots \| K_{l,L}$を生成する.

③ $K_l = K_{l,1} \| K_{l,2} \| \cdots \| K_{l,L}$ に対して $\alpha_l K_l = (\alpha_l K_{l,1}, \alpha_l K_{l,2}, \dots, \alpha_l K_{l,L})$を計算し,$\alpha_l K_l$をランプ型秘密分散法,$\alpha_{l,0}, \dots, \alpha_{l,k-1}$を(k,n)Shamir 法で分散し,キーワード$K_l$に対する分散値集合として$[K_l]_i^{(1)} := (\overline{[\alpha_l K_l]_i}, \overline{[\alpha_{l,0}]_i}, \dots, \overline{[\alpha_{l,k-1}]_i})$を生成する.ただし,$\overline{[\alpha_l K_l]_i}$は以下のように生成する.

$$\overline{[\alpha_l K_l]_i} = \alpha_l K_{l,1} + \alpha_l K_{l,2} x_j + \cdots + \alpha_l K_{l,L} x_j^{L-1} + a_L x_j^L + \cdots + a_{k-1} x_j^{k-1}$$

④ 同様に,$\delta_l, \delta_{l,0}, \dots, \delta_{l,k-1}$をそれぞれ(k,n)Shamir 法で分散し,1 に対する分散値集合として$[1_l]_i^{(1)} := (\overline{[\delta_l]_i}, \overline{[\delta_{l,0}]_i}, \dots, \overline{[\delta_{l,k-1}]_i})$を生成する.

(4) オーナは前記$[K_l]_i^{(1)}$と$[1_l]_i^{(1)}$を$K(D_m)$に対応させて,$[K(D_m)]_i := ([K_l]_i^{(1)}, [1_l]_i^{(1)} | K_l \in D_m)$を生成する.

(5) オーナは各ドキュメント識別子Did_mと鍵Y_mをそれぞれ(k,n)Shamir 法で分散し,$\overline{[Did_m]_i}, \overline{[Y_m]_i}$を生成する.

(6) オーナは n 台のサーバ$S_i(i = 0, \dots, n-1)$にインデックス

$$I = \{(\overline{[Did_1]_i}, [K(D_1)]_i, \overline{[Y_1]_i}), \dots, (\overline{[Did_M]_i}, [K(D_M)]_i, \overline{[Y_M]_i})\}$$

を送信する.

[トラップドア生成:Trapdoor(K')]

(1) 検索者であるユーザは k 個の乱数$\beta_j(j = 0, \dots, k-1)$を生成し,$\beta = \prod_{j=0}^{k-1} \beta_j$を計算する.同様に,k 個の乱数$\eta_j(j = 0, \dots, k-1)$を生成し,$\eta = \prod_{j=0}^{k-1} \alpha_j$を計算する.

(2) ユーザは検索キーワード K'を定められたサイズに応じて L'個に分割し,$K' = K'_1 \| K'_2 \| \cdots \| K'_{L'}$を生成する.

(3) ユーザは$K' = K'_1 \| K'_2 \| \cdots \| K'_{L'}$に対して$\beta K' = (\beta K'_1, \beta K'_2, \dots, \beta K'_{L'})$を計算し,$\beta K'$をランプ型秘密分散法,$\beta_0, \dots, \beta_{k-1}$を(k,n)Shamir 法で分散し,検索キーワード K'に対する分散値集合として$[K']_i^{(1)} := (\overline{[\beta K']_{i,1}}, \dots, \overline{[\beta K']_{i,k-L'+1}}, \overline{[\beta_0]_i}, \dots, \overline{[\beta_{k-1}]_i})$を生成する.ただし,$\beta K'$の各要素をずらしながら以下のような k-L'+1 個の分散情報を生成する.

$$\overline{[\beta K']_{i,1}} = \beta K'_1 + \beta K'_2 x_j + \cdots + \beta K'_{L'} x_j^{L-1} + b_L x_j^L + \cdots + b_{k-1} x_j^{k-1}$$

$$\overline{[\beta K']_{i,2}} = b_0 + \beta K'_1 x_j + \beta K'_2 x_j^2 + \cdots + \beta K'_{L'} x_j^{L'} + b_{L+1} x_j^{L+1} \ldots + b_{k-1} x_j^{k-1}$$

$$\vdots$$

$$\overline{[\beta K']_{i,k-L'+1}} = b_0 + b_1 x_j + \cdots + \beta K'_1 x_j^{k-L'+1} + \beta K'_2 x_j^{k-L'+2} + \cdots + \beta K'_{L'} x_j^{k-1}$$

(4) 同様に,$\eta, \eta_0, \ldots, \eta_{k-1}$をそれぞれ(k,n)Shamir 法で分散し,1 に対する分散値集合として$[1]_i^{(2)} := (\overline{[\eta]_i}, \overline{[\eta_0]_i}, \ldots, \overline{[\eta_{k-1}]_i})$を生成する.

(5) ユーザは k 台のサーバ$S_j (j = 0, \ldots, k-1)$にトラップドア$T_{K'} = ([K']_i^{(1)}, [1]_i^{(2)})$を送信する.

[検索:Search(I, $T_{K'}$)]

(1) k 台のサーバ$S_j (j = 0, \ldots, k-1)$は送られてきた$T_{K'}$と保存されているインデックス I を利用して以下の 2,3 を全ドキュメント識別子$Did_m (m = 1, \ldots, M)$分繰り返す.

(2) k 台のサーバは$[K(D_m)]_j$に対して以下の操作を行う.

① k 台のサーバは$[K_l]_j^{(1)}$と$[1_l]_j^{(2)}$について秘匿乗算を行い,$[K_l]_j^{(2)} := (\overline{[\alpha_l \eta K_l]_j}, \overline{[\alpha_{l,0}\eta_0]_j}, \ldots, \overline{[\alpha_{l,k-1}\eta_{k-1}]_j})$を生成する.

② 以下の③,④の操作を$T_{K'}$中に含まれる$\overline{[\beta K']_{i,1}}, \ldots, \overline{[\beta K']_{i,k-L'+1}}$に対して個々に行う.

③ k 台のサーバは同様に$\overline{[\beta K']_{j,p}}$と$[1_l]_j^{(1)}$について秘匿乗算を行い,$[K']_{j,p}^{(2)} := (\overline{[\delta K_l]_j}, \overline{[\beta_0 \delta_{l,0}]_j}, \ldots, \overline{[\beta_{k-1}\delta_{l,k-1}]_j})(\mathrm{p} = 1, \ldots, \mathrm{k} - \mathrm{L}' + 1)$を生成する.

④ k 台のサーバは$[K_l]_i^{(2)}$と$[K']_{i,p}^{(2)}$について秘匿減算を行い,$[D_{m,l}]_{j,p}^{(1)} := ((\overline{[\gamma_l (K_l - K')]_{j,p}}, \overline{[\gamma_{l,0}]_{j,p}}, \ldots, \overline{[\gamma_{l,k-1}]_{j,p}})$を生成する.

(3) K 台のサーバは受け取ったすべての$(\overline{[\gamma_l (K_l - K')]_{j,p}}$を復元し,復元したときの分散式の項が秘匿減算前に比べ少なくなっていた場合,それに対応する$(\overline{[Did_m]_i}, \overline{[Y_m(K_l)]_j})$をユーザに送信する.

(4) ユーザは,受け取った$\overline{[Did_m]_i}$を復元する.その後,復元したドキュメント識別子に加え,ダミーの識別子をドキュメント辞書$D_{ID} = \{Did_1, \ldots, Did_m\}$の中から任意に選び,サーバ$S_0$に送信する.ただし,この時選択するダミーのドキュメント識別子は復元したドキュメント識別子以外から一定数選ぶ.

(5) サーバS_0は送られてきたドキュメント識別子に対応する(Did_m, C_m)をユーザに送信する.

(6) ユーザは送られてきた$\overline{[Y_m]_j}$を復元してY_mを得,それに対応するC_mを復号してドキュメント$D_m (m \in Did(K_l))$を得る.

5. まとめ

本論文では,入力と出力の閾値が変わらないパスワード付秘密分散法を拡張して検索可能暗号として機能する秘匿検索手法を提案した.また,提案方式を用いた部分一致検索についても検討した.共通鍵暗号方式で議論されるインデックスサイズの削減,検索の効率化,部分一致検索の安全性の検討が今後の課題である.

参考文献

[1] 堀史明,岸本渡. “委任検索可能暗号のマルチユーザ拡張” SCIS2016(2016).
[2]A. Shamir. “How to share a secret”. Communications of the ACM,22,(11),pp.612-613(1979).
[3]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,no9, pp.945-952(1985)
[4]廣田啓一,茂木一男. “ランプ型秘密分散法の中間的な情報漏洩に関する二,三の考察” CSEC34(2006)
[5]M. Ben-Or, S. Goldwasser, and A. Wigderson, “Completeness theorems for noncryptographic fault-tolerant distributed computation”, Proceedings of STOC 1988, ACM Press, pp.1-10,(1988).
[6]神宮武志,岩村恵市. “除算を含む四則演算に適応可能な秘密分散法を用いた秘匿検索手法の提案”. CSEC70(2015)
[7]青井健,神宮武志,岩村恵市. “n<2k-1 における秘匿計算の安全性検討及び非対称秘密分散法との応用”. CSEC74(2016)
[8]A. Bagherzanbi, S. Jarecki, N. Saxena and Y.Lu.: “Password-protected secret sharing”. Proc. Of the 18th ACM Conference on Computer and communications security(CCS 2011)pp.433-444,(2011).
[9]尾形わかは:“情報理論的に安全なパスワード付秘密分散法の安全性と効率化”,SCIS2013(2013).
[10]中原将貴,岩村恵市. “n<2k-1 において実行可能なパスワード付き秘密分散法”. CSS2016(2016)
[11]R. Curtmola, J Garay, S.Kamara, and R.Ostrovsky: “Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions”, In the 13th ACM Conference on Computer and Communications Security(CCS 2006), pp.79-88, (2006).
[12]平野貴人,河合豊,太田和夫,岩本貢: “共通鍵暗号型の秘匿部分一致検索（その１）”, SCIS2016(2016).
[13]川合豊, 平野貴人. “部分一致検索可能暗号の効率化に向けて”. SCIS2015(2015)
[14]S. Kamara, and C.Papamanthou, “Parallel and Dynamic Searchable Symmetric Encryption,” In Financial Cryptography(FC2013), (2013)
[15]早坂健一郎,河合豊,小関義博,平野貴人,岩本貢,太田和夫“検索クエリからの情報漏洩を削減した効率的な共通鍵型検索可能暗号”,SCIS2017(2017)
[16]山岡裕司,牛田芽生恵,伊藤孝一: “頻度分析を k-匿名性で緩和する検索可能共通鍵暗号”, CSS2015(2015).
[17]吉野雅之,長沼健,佐藤尚宜: “DB 向け検索可能暗号方式の検討(2)”,SCIS2011(2011)
[18]渡邊尊司,山本博章: “階層型ブルームフィルタを用いた暗号化検索法の改良”, CSS2014(2014).
[19]佐藤尋時,大瀧保広. “Forward Privacy を考慮した動的な検索可能暗号”.SCIS2015(2015)