L-008

# 公開鍵検索可能暗号に適したブルームフィルタの検討
# Constructions of Bloom Filter for PEKS

柴山 綸太郎 †
Rintaro Shibayama

土井 洋 †
Hiroshi Doi

## 1. はじめに

近年，クラウド (外部のサーバ) 上にデータを保存し，必要な時に必要なデータをクライアント (PC など) にダウンロードして利用する形が増加している．保存するデータが機密データの場合は暗号化して保存することになるが，保存されている大量のデータから必要なデータをサーバに検索させることで，通信量やクライアントの計算量を削減できる．このような用途に利用できるものとして，検索可能暗号がある．検索可能暗号は共通鍵検索可能暗号と公開鍵検索可能暗号に大別できる．共通鍵検索可能暗号では，データを保存するエンティティと利用するエンティティが事前に秘密情報を共有する．公開鍵検索可能暗号では事前の秘密情報の共有は不要であり，どのエンティティでもデータを暗号化して保存できる [1]．また，空間効率よく検索を行うためのデータ構造としてブルームフィルタ [5] が知られており，ブルームフィルタを用いた検索可能暗号として，[2], [3], [4], [8] などが知られている．

これらのうち [2] は結果秘匿 (サーバに対して検索結果の秘匿) という高い安全性を有しており，暗号文同士の加算及び乗算が任意の回数可能である共通鍵完全準同型暗号を用いて構成される．なお，結果秘匿を実現するために，従来の検索可能暗号 ([1], [3], [4]) とは異なり，サーバとの通信回数が増えざるを得ない．

結果秘匿を実現することはプライバシ保護の観点等で有用であるが，結果秘匿を実現する検索可能暗号についてはあまり研究がなされていない．本研究では，結果秘匿機能を実現する (小嶋ら [2] のモデルと同等のモデルとなる) ブルームフィルタを利用可能な公開鍵検索可能暗号の設計を目標とする．この際，暗号文同士の加算及び乗算が可能な準同型暗号を利用するが，乗算回数が少なくて済むようにブルームフィルタの検討を行う．

### 1.1. 貢献

提案手法は小嶋らの手法 [2] をベースとした公開鍵検索可能暗号であり，以下を達成している．

**暗号文サイズの固定化**　多くの公開鍵検索可能暗号方式，例えば [1] においてはキーワード数の増加に比例して暗号文サイズも増加する．ブルームフィルタには (上限はあるが) 複数キーワードを登録可能であり，キーワード数が増加しても暗号文長は増加しない．

**要素秘匿・結果秘匿の達成**　[2] と同様，暗号文に紐付いたキーワードがサーバに漏れないこと (要素秘匿) と，どの暗号文が検索クエリと一致したのかがサーバに漏れないこと (結果秘匿) という安全性を達成している．

**様々な準同型暗号方式を利用可能な構成**　[2] は (任意の回数の乗算が可能な) 完全準同型暗号を利用しているが，提案手法では乗算が一度だけ可能で，加算は任意の回数可能な準同型暗号を利用することができる．

† 情報セキュリティ大学院大学
IISEC, Kanagawa, Yokohama 221–0835, Japan

## 2. 準備

本章では，本稿で使用する表記や要素技術の定義を行う．

### 2.1. 想定モデル

提案手法は結果秘匿可能な公開鍵検索可能暗号であり，エンティティは図 1 に示すように複数の送信者，単一の受信者，そして (完全には信頼できない) サーバの三者である．

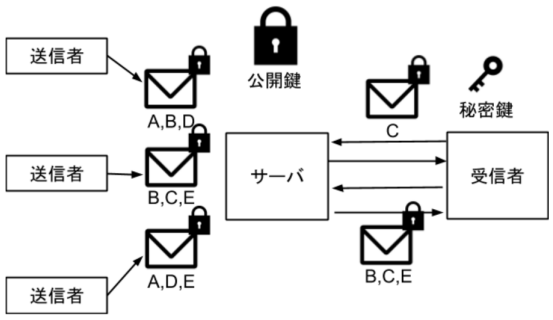


**図 1**　想定モデルとエンティティ

送信者は公開鍵を用いてデータの暗号文と，データに紐づくキーワードの暗号文 (検索可能暗号文) を生成し，これらをサーバに保管する．受信者はキーワードからトラップドアを生成し，サーバに対しトラップドアを送る．サーバはトラップドアと保管している検索可能暗号文をもとに検索アルゴリズムを実行し，実行結果を受信者に送信する．受信者は実行結果をもとに，サーバからデータの暗号文を得る．なお，エンティティ間で秘密裏の情報共有はしない．

なお，検索可能暗号文の生成やトラップドアの生成時に，ブルームフィルタの利用を可能とする．特に明示しない限り，検索可能暗号文の生成時には複数のキーワードを入力し，トラップドア生成時にはひとつのキーワードを入力するものとする．

### 2.2. 記法

$x$ を入力としてアルゴリズム $Alg$ を実行し $A$ を出力することを $A \leftarrow Alg(x)$ と表記する．集合 $S$ から要素 $s$ を一様ランダムに選ぶことを $s \xleftarrow{u} S$ と表記する．変数 $V$ に $v$ を代入することを $V \leftarrow v$ と表記する．$\{0,1\}^n$ を長さ $n$ のビット列全体，特に $\{0,1\}^*$ と書いたときは任意の長さのビット列全体とする．$\{\mathbb{Z}_q\}^m$ を $\mathbb{Z}_q$ の元を要素とする長さ $m$ の数列全体とする．$b^n$ を $n$ 個の $b \in \{0,1\}$ からなるビット列とする．$N$ を数列，またはビット列と

するとき，$N[i]$ を $N$ の $i$ 番目の要素とする．$N$ を要素数 $x$ のビット列，または数列とするとき，$\Sigma_x N$ を $N$ の要素の和とする．$N_1, N_2 \in \{0,1\}^n$ とするとき，$N_1 \cup N_2$ を $N_1$ と $N_2$ のビットごとの論理和，$N_1 \times N_2$ を $N_1$ と $N_2$ のビットごとの論理積とする．また，$NOT(N)$ を $N$ の各ビットの 1 と 0 を反転させたビット列とする．$\mathcal{F}$ をハッシュ関数族，$\mathcal{M}$ を平文空間，$\mathcal{C}$ を暗号文空間とする．最後に，ビット列の包含関係の定義を次のように定義する．$N_1, N_2 \in \{0,1\}^n$ のとき, 任意の $0 \leq i < n$ に対して $N_2[i] = 1$ なら $N_1[i] = 1$ が成り立つとき，“$N_2$ は $N_1$ に含まれる”という．

### 2.3. 既存研究

小嶋らが 2016 年に発表した [2] において，共通鍵完全準同型暗号とブルームフィルタを用いることで安全なブルームフィルタを構成可能であることが示されている．[2] においては共通鍵完全準同型暗号を用いてブルームフィルタの各ビットを暗号化し，ビット数に比例するサイズの検索可能暗号文を生成することでブルームフィルタを用いた検索を達成している．[2] は公開鍵検索可能暗号方式にも適用可能である．また，Boneh らの手法 [1] と異なり，[2] は検索結果をサーバに対し秘匿するモデルである．したがって [1] と [2] を比較すると，[2] はサーバと受信者間の対話が 1 回増えているものの，検索結果をサーバから秘匿することが可能である．

### 2.4. ブルームフィルタ

ブルームフィルタは 1970 年に Bloom が提案した [5] データ構造である．ハッシュ関数を用いて固定長のビット列に複数のキーワードを登録することができるが，登録されていないキーワードについて検索を行った際に一致と誤判定してしまう偽陽性による誤検出の可能性がある．

ブルームフィルタは以下の 4 つのアルゴリズムで構成され，$BFSetup$ と $Trapdoor$ を受信者が，$BuildBloomFilter$ を送信者が，そして $JudgeBloomFilter$ をサーバが実行する．ブルームフィルタのビット数は $n$ ビットとする．

$BFSetup(l)$

$for\ i \leftarrow 1\ to\ m$
$f_i \xleftarrow{u} \mathcal{F}$
$end\ for$
$output : \{f_i\}(i = 1, ..., m)$

セキュリティパラメータ $l$ を入力し，ハッシュ関数族 $\mathcal{F}$ から $m$ 個のハッシュ関数 $f_i$ $\{0,1\}^* \rightarrow \mathbb{Z}_n$ $(i = 1, ..., m)$ を出力する確率的アルゴリズムである．

$Trapdoor(w \in \{0,1\}^*)$

$Td \leftarrow 0^n$
$for\ i \leftarrow 1\ to\ m\ do$
$j \leftarrow f_i(w)$
$Td[j] \leftarrow 1$
$end\ for$
$output : Td$

検索キーワード $w$ を入力し，検索用トラップドア $Td \in \{0,1\}^n$ を出力する確定的アルゴリズムである．

$BuildBloomFilter(w_s \in (\{0,1\}^*)^s)$

$BF \leftarrow 0^n$
$for\ w \leftarrow w_s$
$BF_w \leftarrow 0^n$
$for\ i \leftarrow 1\ to\ m\ do$
$j \leftarrow f_i(w)$
$BF_w[j] \leftarrow 1$
$end\ for$
$BF \leftarrow BF \cup BF_w$
$end\ for$
$output : BF$

紐づけたい $s$ 個のキーワード $w_s$ を入力し，ブルームフィルタ $BF \in \{0,1\}^n$ を出力する確定的アルゴリズムである．

$JudgeBloomFilter(Td, BF)$

$for\ i \leftarrow 0\ to\ n-1$
$if\ Td[i] = 1\ and\ BF[i] = 0$
$output : \perp$
$end\ for$
$output : 1$

$Td, BF$ を入力し，$Td$ が $BF$ に含まれていれば 1 を，そうでなければ $\perp$ を出力する確定的アルゴリズムである．

ブルームフィルタを用いた登録・検索は上記の 4 アルゴリズムを用いて以下のように進行する．

**登録**

1. 受信者は $BFSetup$ を実行し，$m$ 個のハッシュ関数 $f_1, ..., f_m$ を生成，送信者と共有する．
2. 送信者はデータに紐づけたい $s$ 個のキーワード $w_i (i = 1, ..., s)$ について $BuildBloomFilter$ を実行しブルームフィルタ $BF$ を構築，サーバに送信する．

**検索**

1. 受信者は検索したいキーワード $w$ について $Trapdoor$ を実行し $Td$ を生成，サーバに送信する．
2. サーバは届いた $Td$ とサーバに格納された全ての $BF$ について $JudgeBloomFilter$ を実行し $1\ or\ \perp$ を計算，1 と判定されたものを受信者に送信する．

ブルームフィルタ自体には登録されたキーワードを秘匿する機能はないため，共通鍵検索可能暗号方式でブルームフィルタを利用する際には $BFSetup$ で生成したハッシュ関数を秘密鍵として用いる構成が取られる．また，[2] では，ブルームフィルタ及びトラップドアの各ビットを完全準同型暗号方式を用いて暗号化し，各ビットの論理演算を行う手法を取っている．

### 2.5. 準同型暗号

準同型暗号とは暗号文を復号することなく，暗号文のまま平文についての演算が可能な性質をもつ暗号方式である．個人情報を含むデータを暗号化したまま演算することができ，生体認証や遺伝子医療などさまざまな分野での応用が研究されている [7]．提案手法で用いる準同型暗号は以下の 8 つのアルゴリズムで構成される．以下，平文空間 $\mathcal{M}$ を $\mathbb{Z}_q$ とする．

$Keygen(\tau)$

セキュリティパラメータ $\tau$ を入力とし，公開鍵 $PK$ 及び秘密鍵 $SK$ を出力する確率的アルゴリズムである．

$Enc_1(PK, M)$
: $PK$，平文 $M \in \mathbb{Z}_q$ を入力し，暗号文 $c \in \mathcal{C}_1$ を出力する確率的アルゴリズムである．

$Enc_2(PK, M)$
: $PK$，平文 $M \in \mathbb{Z}_q$ を入力し，暗号文 $C \in \mathcal{C}_2$ を出力する確率的アルゴリズムである．

$Mult(c_1, c_2)$
: $c_1, c_2 \in \mathcal{C}_1$ を入力し，平文について乗算がなされた新たな暗号文 $C_{Mult} \in \mathcal{C}_2$ を出力する確率的アルゴリズムである．

$Add_1(c_1, c_2)$
: $c_1, c_2 \in \mathcal{C}_1$ を入力し，平文について加算がなされた新たな暗号文 $c_{Add} \in \mathcal{C}_1$ を出力する確率的アルゴリズムである．

$Add_2(C_1, C_2)$
: $C_1, C_2 \in \mathcal{C}_2$ を入力し，平文について加算がなされた新たな暗号文 $C_{Add} \in \mathcal{C}_2$ を出力する確率的アルゴリズムである．

$Dec_1(SK, c)$
: $SK, c \in \mathcal{C}_1$ を入力し，復号結果 $M' \in \mathbb{Z}_q$ を出力する確定的アルゴリズムである．

$Dec_2(SK, C)$
: $SK, C \in \mathcal{C}_2$ を入力し，復号結果 $M' \in \mathbb{Z}_q$ を出力する確定的アルゴリズムである．

提案手法では任意の回数の加算，一度のみの乗算が可能な公開鍵準同型暗号方式 (以下，$M_1$ 準同型暗号と呼ぶこととする) を用いる．このような性質をもつ準同型暗号には例えば BGN 暗号 [6] などが挙げられる．上記の準同型暗号における加法・乗法準同型性を以下のように定義する．

**加法準同型性**

任意の $m_1, m_2 \in \mathbb{Z}_q$ に対して

$$\begin{aligned} & Dec_1(Add_1(Enc_1(m_1), Enc_1(m_2))) \\ = \ & Dec_1(Enc_1(m_1 + m_2)) \quad , \\ & Dec_2(Add_2(Enc_2(m_1), Enc_2(m_2))) \\ = \ & Dec_2(Enc_2(m_1 + m_2)) \end{aligned}$$

を満たす．

**乗法準同型性**

任意の $m_1, m_2 \in \mathbb{Z}_q$ に対して

$$\begin{aligned} & Dec_2(Mult(Enc_2(m_1), Enc_2(m_2))) \\ = \ & Dec_2(Enc_2(m_1 \times m_2)) \end{aligned}$$

を満たす．

BGN 暗号などでは，乗算を一度も行っていない暗号文と乗算を一度行った暗号文の暗号文空間は異なり，両者の加算は行えない．またこの 2 つの暗号文を同じ復号アルゴリズムでは復号できないため，$Enc, Add, Dec$ についてアルゴリズムが 2 つずつ必要となる．

## 3. ブルームフィルタを用いた共通鍵検索可能暗号

本章では小嶋らの先行法 [2] におけるブルームフィルタを用いた共通鍵検索可能暗号の登録・検索手順を紹介する．[2] のアルゴリズムではトラップドアの生成にもブルームフィルタを利用しており，ブルームフィルタ及びトラップドアの各 $n$ ビット毎に共通鍵完全準同型暗号を用いて暗号化している．そのため，暗号化ブルームフィルタと暗号化トラップドアは各々 $n$ 個の暗号文から成る．そしてサーバが暗号化ブルームフィルタと暗号化トラップドアの各ビットごとに所定の演算を行う．具体的な手順を以下に示す．なお，$Enc, Dec$ は共通鍵完全準同型暗号の暗号化と復号アルゴリズムである．その他，一部の記号については [2] を参照のこと．

**登録**

(1) 送信者は登録したい 1 つ以上のキーワード $w_1, ..., w_s$ から $n$ ビットのブルームフィルタ $BF$ を生成する．

(2) 送信者は完全準同型暗号の暗号化アルゴリズム $Enc$ を $n$ 回起動し $EBF \leftarrow \{Enc(BF[i])\}_{i=0,...,n-1}$ とし，$n$ 個の暗号文からなる $EBF$ をサーバに保管する．

**検索**

(1) 受信者は検索したい 1 つのキーワード $w$ から $n$ ビットのトラップドア $Td$ を生成する．

(2) 受信者は完全準同型暗号の暗号化アルゴリズム $Enc$ を $n$ 回起動し $ETd \leftarrow \{Enc(BF[i])\}_{i=0,...,n-1}$ とし，$ETd$ をサーバに送信する．

(3) サーバは送信されている全ての $EBF$ について

$$\begin{aligned} z &\leftarrow \Sigma_n(AND_n(EBF, ETd)), \\ E_{mask} &\leftarrow \Sigma_n(ETd), \\ Eb &\leftarrow E_{mask} - z \end{aligned}$$

を計算し，サーバに保管されている $EBF$ の数だけの $Eb$ を受信者に送信する．ただし $AND_n$ は 2 つの暗号文の各々の平文をビットごとに $AND$ 演算した結果の暗号文を出力するアルゴリズムである．

(4) 受信者は自身のもつ秘密鍵にて完全準同型暗号の復号アルゴリズム $Dec$ を起動し $b \leftarrow Dec(Eb)$ を得る．

(5) 受信者は b が 0 であれば一致，それ以外のものを不一致と判定する．

手順 (3) の計算においては完全準同型暗号を利用するため，値についての情報がサーバに漏れることはない．一致判定には以下の定理を利用している．

**定理 1** $Td$ が $BF$ に含まれている場合，$Td = BF \times Td$ である．含まれていない場合，$\Sigma_n(Td) > \Sigma_n(Td \times BF)$ である．

**証明** 定理前半の証明は [2] にある．定理後半について，$Td$ が $BF$ に含まれないとすると，ビット列の包含関係の定義から $Td[i] = 1, BF[i] = 0$ なるビットのアドレス $i$ が存在する．そのアドレス $i$ においては $Td[i] \times BF[i] = 0$ である．よって $\Sigma_n(Td) > \Sigma_n(Td \times BF)$ となる．

## 4. $M_1$ 準同型暗号を用いた公開鍵検索可能暗号

本章ではブルームフィルタと $M_1$ 準同型暗号を組み合わせた公開鍵検索可能暗号の構成法について述べる．提案手法でも [2] と同様，ブルームフィルタ及びトラップドアのビット列を分割し $M_1$ 準同型暗号を用いて暗号化する．提案手法では $n$ ビットのブルームフィルタを利用するとき，必要な暗号文の数は $n$ 個となる．以下の手順中の $KeyGen, Enc_1, Dec_2, Add_2, Mult$ は 2.5 節で定義したものである．

登録

(1) 送信者は登録したい 1 つ以上のキーワード $w_1, ..., w_s$ について $BF' \leftarrow BuildBloomFilter(w_i)(i = 1, ..., s)$ としブルームフィルタ $BF' \in \{0,1\}^n$ を出力する．

(2) 送信者は $BF \leftarrow NOT(BF')$ とする．

(3) 送信者は $Enc$ を $n$ 回起動し $EBF[i] \leftarrow Enc_1(BF[i])$ $(i = 0, ..., n-1)$ とし $n$ 個の暗号文から成る $EBF$ をサーバに保管する．

検索

(1) 受信者は検索したい 1 つのキーワード $w$ について $Trapdoor(w)$ を起動し検索用トラップドア $Td \in \{0,1\}^n$ を出力する．

(2) 受信者は $Enc_1$ を $n$ 回起動し $ETd[i] \leftarrow Enc_1(Td[i])$ とし $n$ 個の暗号文から成る $ETd$ をサーバに送信する．

(3) サーバは保管されている全ての $\{EBF_j\}$ について，$Mult$ を $n$ 回ずつ起動し $Eb_j[i] \leftarrow Mult(EBF_j[i], ETd[i])$ を計算する．

(4) サーバは $Eb_j[i]$ について，$Add_2$ を $n-1$ 回起動し $Eb_j \leftarrow \Sigma(Eb_j[i])$ を計算し $\{Eb_j\}$ を受信者に送信する．

(5) 受信者は自身のもつ秘密鍵にて $Dec_2$ を起動し $b \leftarrow Dec_2(Eb_j)$ を計算する．

(6) 受信者は $b = 0$ ならば一致と判定する．それ以外の数値であれば不一致と判定する．

[2] と同様，検索手順 (3)(4) において値についての情報がサーバに漏れることはない．検索手順 (6) の判定には以下の定理 2 を利用している．

**定理 2** $Td$ が $BF$ に含まれている場合，$Td \times NOT(BF) = 0^n$ である．

**証明** $Td$ が $BF$ に含まれているならば，$Td[i] = 1$ となる任意の $i$ について $BF[i] = 1$ である．よって $Td[i] = 1$ ならば $NOT(BF[i]) = 0$ であり，$Td[i] \times NOT(BF[i]) = 0$ である．一方，$Td[i] = 0$ となる任意の $i$ について，$Td[i] \times NOT(BF[i]) = 0 \times NOT(BF[i]) = 0$ である．よって $Td \times NOT(BF) = 0^n$ となる．

**例 1** 表 1 は $n = 8$ としたときの提案手法の実行例である．キーワードが含まれている際には $b$ は 0 となることがわかる．一方で不一致の際には必ず異なる数値が現れるため，キーワードの一致判定が可能である．

**表 1** ブルームフィルタの一致判定

| | $BF$ | $Td_A$(一致) | $Td_B$(不一致) |
|---|---|---|---|
| $BF$ | (10111110) | (00101100) | (01010011) |
| $NBF$ | (01000001) | - | - |
| $Td \times NBF$ | - | (00000000) | (01000001) |
| $b$ | - | 0 | 2 |

## 5. 提案手法の機能

[2] と同様，提案手法もブルームフィルタを用いることで様々な機能をもつ．本章では，提案手法がもつ機能について述べる．

### 5.1. 複数単語の一括検索

ブルームフィルタの性質を用いることで，[2] と同様に複数単語の一括 AND 検索に対応した検索用トラップドアを生成可能である．以下のアルゴリズム $ANDTrapdoor$ はキーワード $w_1, ..., w_s$ について生成されたトラップドア $Td_1, ..., Td_s$ を入力とし，AND 検索を行うためのトラップドア $Td_{AND}$ を生成するアルゴリズムである．

$ANDTrapdoor(Td_1, ..., Td_s)$
$Td_{AND} \leftarrow 0^n$
$for\ i \leftarrow 1\ to\ s$
$Td_{AND} \leftarrow Td_{AND} \cup Td_i$
$end\ for$
$output : Td_{AND}$

### 5.2.EBF の和集合

$BF$ の和集合 $BF_{Join}$ とは，

$BF_1 \leftarrow BuildBloomFilter(w_s)$,
$BF_2 \leftarrow BuildBloomFilter(w_t)$

としたとき，キーワード $w' = w_s \cup w_t$ について生成された $BF_{Join} \leftarrow BuildBloomFilter(w')$ となるブルームフィルタである．

提案手法では $BF_1$ と $BF_2$ をそれぞれ暗号化した $EBF_1$ と $EBF_2$ から $EBF_{Join} = EBF_{w'}$ を生成することができる．受信者は $EBF_{Join}$ を用いることで，複数の $EBF$ の中に 1 つ以上登録されているキーワードを検索することができる．

#### 5.2.1 提案手法における EBF の和集合

提案手法において $EBF$ の和集合をとる際は送信する $Eb$ の数は多くなるものの，$Mult$ を用いる必要がない．アルゴリズム $Join$ を以下のように定義する．

**Join** $(EBF_1, EBF_2)$
$for\ i \leftarrow 0\ to\ n-1$
$EBF_{join}[i] \leftarrow Add_1(EBF_1[i], EBF_2[i])$
$end\ for$
$output : \{EBF_{join}[i]\}_{i=0,...,n-1}$

上記のアルゴリズム $Join$ を用いて，受信者は $s$ 個の $EBF$ の和集合から以下の手順でキーワードを検索可能である．

(1) 受信者はサーバに対し，和集合をとりたい暗号文 $EBF_1, ..., EBF_s$ を指定する．

(2) サーバは $EBF_{Join} \leftarrow Join(EBF_s, ..., Join(EBF_3, Join(EBF_2, EBF_1)))...)))$ を計算する．

(3) 受信者は和集合について検索したいキーワード $w$ について，$Td \leftarrow Trapdoor(w)$ を出力する．

(4) 受信者は $Enc_1$ を $n$ 回起動し $ETd[i] \leftarrow Enc_1(Td[i])$ とし，$ETd$ をサーバに送信する．

(5) サーバは $Eb[i] = Add_1(ETd[i], EBF_{Join}[i])$ を計算し，受信者へ $\{Eb[i]\}_{i=0,...,n-1}$ を送信する．

(6) 受信者は自らの持つ秘密鍵で $b[i] \leftarrow Dec(Eb[i])$ を計算し，全ての $b[i]$ が $b[i] \neq s+1$ であれば 1 を，そうでなければ $\perp$ を出力する．

この方法でブルームフィルタの和集合に対して検索ができる理由は以下の定理 3 による．

**定理 3** $Td$ が $BF_j (j = 1, ..., s)$ のいずれにも含まれな

表 2　評価

| 名称 | 要素秘匿 | 結果秘匿 | $BF$ の和集合 | $BF$ の共通集合 | 暗号方式 |
|---|---|---|---|---|---|
| [1] | 有 | 無 | 無 | 無 | - |
| [2] | 有 | 有 | 有 | 有 | 完全準同型暗号 |
| 提案手法 | 有 | 有 | 有* | 有 | $M_1$ 準同型暗号 |

い場合，ある $i$ が存在し，$Td[i] + NOT(BF_1[i]) + ... + NOT(BF_s[i]) = s + 1$ となる．

**証明**　ビット列の包含関係の定義から，$Td$ がある $BF$ に含まれているとき $Td[i] = 1$ となる全ての $i$ について $BF[i] = 1$ である．ここで $BF_j(j = 1, ..., s)$ について，その中に 1 つでも $Td$ を含むものがある場合を考える．$BF_j$ のうち $Td$ を含むブルームフィルタのひとつを $BF_k$ としたとき，ある $i$ が存在し $Td[i] = BF_k[i] = 1$ となる．よって $Td$ が $BF_j(j = 1, ..., s)$ のいずれにもに含まれていない場合，ある $i$ が存在し $Td[i] = 1$ かつ $BF_1[i] = 0, ..., BF_j[i] = 0$ となる．よって $Td[i] + NOT(BF_1[i]) + ... + NOT(BF_s[i]) = s + 1$ であるような $i$ が存在する．

**例 2**　以下の表 3 は $BF_1 = (100101), BF_2 = (001010), BF_3 = (011000), Td = (100100)$ としたときの $EBF_{Join}$ の計算の例である．$BF_1$ のみが $Td$ を含むため，$BF_1$ と $BF_2$，$BF_1$ と $BF_3$ の和集合をとった場合は一致と判定される．$BF_2$ と $BF_3$ の和集合で計算した場合には $Td + EBF_{Join}$ に $s + 1 = 3$ となるような要素が存在し，不一致であることがわかる．

表 3　$BF$ の和集合

| | $BF_1, BF_2$ | $BF_2, BF_3$ | $BF_1, BF_3$ |
|---|---|---|---|
| $BF_{Join}$ | (121111) | (210212) | (111121) |
| $Td + BF_{Join}$ | (221211) | (310312) | (211221) |

### 5.3.EBF の共通集合

$BF$ の共通集合 $BF_{Inte}$ とは，

$BF_1 \leftarrow BuildBloomFilter(w_s)$,
$BF_2 \leftarrow BuildBloomFilter(w_t)$

としたとき，キーワード $w' = w_s \cap w_t$ について生成された $BF_{Inte} \leftarrow BuildBloomFilter(w')$ となるブルームフィルタである．

提案手法では $BF_1$ と $BF_2$ をそれぞれ暗号化した $EBF_1$ と $EBF_2$ から暗号化ブルームフィルタの共通集合 $EBF_{Inte} = EBF_{w'}$ を生成することができる．受信者は $EBF_{Inte}$ を用いることで，複数の $EBF$ のいずれにも登録されているキーワードを検索することができる．

#### 5.3.1　提案手法における EBF の共通集合

提案手法において $EBF$ の共通集合をとる際は送信する $Eb$ の数は 1 つであるが，$Mult$ を用いる必要がある．アルゴリズム $Inte$ を以下のように定義する．

**Inte**　$(EBF_1, EBF_2)$
　$for\ i \leftarrow 0\ to\ n-1$
　　$EBF_{Inte}[i] \leftarrow Add_1(EBF_1[i], EBF_2[i])$
　$end\ for$
　$output : \{EBF_{Inte}[i]\}_{i=0,...,n-1}$

上記のアルゴリズム $Inte$ を用いて，受信者は $EBF$ の共通集合から以下の手順でキーワードを検索可能である．

(1) 受信者はサーバに対し，共通集合をとりたい暗号文 $EBF_1, ..., EBF_s$ を指定する．
(2) サーバは $EBF_{Inte} \leftarrow Inte(EBF_s, ..., Inte(EBF_3, Inte(EBF_2, EBF_1)))...)))$ を計算する．
(3) 受信者は共通集合について検索したいキーワード $w$ について，$Td \leftarrow Trapdoor(w)$ を出力する．
(4) 受信者は $Td$ を $ETd[i] \leftarrow Enc_1(Td[i])$ とし，$ETd$ をサーバに送信する．
(5) サーバは $EBF_{Inte}$ について，$Mult$ を $n$ 回起動し $Eb_i \leftarrow Mult(EBF_{Inte}[i], ETd[i])$ を計算する．
(6) サーバは $Eb_i$ について，$Add_2$ を $n-1$ 回起動し $Eb \leftarrow \Sigma(Eb_i)$ を計算し，$Eb$ を受信者に送信する．
(7) 受信者は自らの持つ秘密鍵で $b \leftarrow Dec(Eb)$ を計算し，$b = 0$ であれば 1 を，そうでなければ $\perp$ を出力する．

この方法で共通集合が取れる理由は定理 2 による．

## 6. 評価

他手法との比較結果を表 2 へとまとめる．要素秘匿，結果秘匿，ブルームフィルタの和集合・共通集合，そして暗号方式の 4 点において比較を行った．提案手法は要素秘匿・結果秘匿を達成し $M_1$ 準同型暗号方式で実現できるものの，BF の和集合については [2] と比較したときに通信量と受信者の計算量が大きくなることが課題として挙げられる．

## 7. 今後の課題

提案手法の安全性は用いる準同型暗号の安全性に帰着されるものと考えているが，フォーマルな安全性の証明が今後の課題である．. 併せて詳細な性能評価も課題である．また，加法準同型性のみをもつ準同型暗号 ($M_0$ 準同型暗号) においての公開鍵検索可能暗号の構成，そして部分一致検索への対応も模索していきたい．

### 参考文献


[1] Boneh, D., Crescenzo, G., Ostrovsky, R., and Persiano, G.: Public Key Encryption with Keyword Search, EUROCRYPT 2004, pp.506-522(2004).
[2] 小嶋 陸大，品川 和雅，金山 直樹，西出 隆志，“共通鍵完全準同型暗号を用いた安全なブルームフィルタ”，SCIS2016，2A2-3(2016).
[3] 佐保 航輝，油田 健太郎，山場 久昭，保田 真一郎，朴 美娘，岡崎 直宣，“検索可能暗号におけるブルームフィルタを応用した検索方式の提案”，研究報告コンピュータセキュリティ (CSEC)2015.8，pp.1-7(2015).
[4] 菅 孝徳，西出 隆志，櫻井 幸一，“ブルームフィルタを用いた検索自由度の高い検索可能暗号の設計と実装評価”，研究報告コンピュータセキュリティ (CSEC)2011.20，1-6(2011).
[5] Bloom, B, H.:Space/time trade-offs in hash coding with allowable errors, Commun.ACM, vol.13, no.7, pp.422-

426(1970).
[6] Boneh, D., Goh, E., and Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts, TCC 2005: Theory of Cryptography, pp.325-341(2005).
[7] 下山 武司. “暗号を解かずにデータ処理 -準同型暗号の仕組みと産業応用”, 情報処理, Vol.57, No.1, pp.44-50(2015).
[8] Goh, E.:Secure Indexes, Cryptology ePrint Archive, Report 2003/216(2003).